A PROCEDURE FOR THE COMPUTATION OF SEA SURFACE ADVECTION VELOCITIES FROM SATELLITE THERMAL BAND IMAGERY, WITH APPLICATIONS TO THE SOUTH EAST ATLANTIC OCEAN.

by

JOHANNES JACOBUS AGENBAG

Submitted in fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Faculty of Science at the
UNIVERSITY OF CAPE TOWN

June 1992

Supervisor : Dr. F.A. Shillington

ACKNOWLEDGEMENTS

This study was undertaken at the Sea Fisheries Research Institute (SFRI), Cape Town, and I thank the Director, Dr. L.V.Shannon for permission to write this thesis and for creating the opportunity for me to do so. I also wish to thank my supervisor Dr. F.A.Shillington of the University of Cape Town for the academic guidance, criticisms and support received from him.

All my colleagues in the Physical/Chemical Section of the SFRI has contributed in some manner or other towards this thesis and I wish to express my sincerest appreciation for their help. In particular I thank Dr. Shannon and Geoff Bailey, as former and present heads of the section for their encouragement and support; Grev Nelson for his advice on those occasions when the mathematics were beyond me and Alan Boyd and Sybrand Mostert for proof reading the manuscript. Alan is also thanked for his criticisms and helpful advice - his contribution towards improving the quality of the manuscript has been substantial and is greatly appreciated.

A useful fringe benefit derived from producing this thesis is that I learned to use the Institute's word processing package. I thank Iona Styles and Chris Duncombe Rae for their patient tutoring and Iona for performing the difficult typing in Chapter 2. Tony van Dalsen and his staff produced much of the artwork in the midst of an already very busy schedule - their efforts are also much appreciated.

Finally I want to express my deepest gratitude towards my wife for her encouragement and for shouldering the lion's share of the domestic duties during the past three years and I thank the Lord for giving me the strength and wisdom to carry through this task.

ABSTRACT

"A PROCEDURE FOR THE COMPUTATION OF SEA SURFACE ADVECTION VELOCITIES FROM SATELLITE THERMAL BAND IMAGERY, WITH APPLICATIONS TO THE SOUTH EAST ATLANTIC OCEAN"

by Johannes Jacobus Agenbag, Sea Fisheries Research Institute, Private Bag X2, Roggebaai, Cape Town, South Africa.
June 1992.

The research was carried out with a view to developing a procedure for the computation of sea surface advection velocities from pairs of NOAA AVHRR infrared images. The procedure was designed for application to the oceanic regions around South Africa and cognisance had to be taken of restrictions imposed by the specific oceanographic conditions, availability of satellite data, as well as the capabilities of the image processing system used.

As a first step, a set of image navigation algorithms was developed, based on elliptical orbit and ellipsoidal earth models. Orbit parameters were obtained from TBUS-bulletins and one or more ground reference points had to be identified on each. The navigation algorithms were then used to develop a procedure for the geometric transformation of images to a Mercator map projection. The transformation procedure was evaluated through use of test-images and the results indicated that the maximum errors which could be expected in the computation of advection vectors were 4-5 cm/s in the north/south velocity component and 6-7 cm/s in the east/west component if two images, 12 hours apart in time, were used for the vector computation.

An automatic feature tracking method was tested as a means for computing advection velocities but was found to be unsatisfactory. As a result, a 'semi-automated' procedure was developed. This process is essentially a manual (point-wise) feature tracking procedure into which the template matching technique which formed the basis of automated procedures, was incorporated as a labour saving device. Tests indicated a time saving of 20-40 % on the manual procedure and more rapid computation than with the automated procedure.

The feature tracking procedure was applied to three sets of AVHRR images of the South East Atlantic. To assess the precision of the vector computation procedure, two independent vector sets were computed. A comparison of the two sets indicated that the root-mean-square deviation in vector magnitude (speed) was about 6-8 cm/s and in the vector direction, about 31° (12° if very small vectors ≤ 6 cm/s are excluded). The computed vectors compared very well with reported results from conventional methods. The derived vector fields also provide the first really detailed description of surface currents in the sea off South Africa: eg. on the flow field in the southern Benguela Current, the circulation associated with Agulhas Current rings, and advective influences on the transport of fish eggs and larvae from the spawning grounds on the Agulhas Bank to the favoured recruitment area off the West Coast.

TABLE OF CONTENTS

1. INTRODUCTION

A brief history of surface current measurements in South African waters.

Until as recently as the late 1970s, ocean currents around the South African coasts were determined from estimates of ship's drift (Clowes 1954) or the analysis of water properties such as salinity, density and dissolved oxygen (Darbyshire 1963, Shannon 1966, Visser 1969, De Decker 1970, Bang 1976). Drift cards were also extensively used and played an important role in describing nearshore surface advection patterns along the West- and South Coast regions (Duncan and Nell 1969) as well as the basin-wide circulation in the South East- and South Atlantic (Shannon *et al.* 1973). Over a period of time, data obtained from these sources helped to build up an understanding of the general flow rates in the area but the conceptual picture obtained was over simplified and provided little information on the detail of the flow pattern.

Radar- and radio-tracked surface drogues employed during the early 1980s, provided more detailed, Lagrangian descriptions of surface currents, but over limited geographical areas (Boyd 1983, Holden 1985). High spatial resolution was also on occasion achieved through the use of ship-deployed current meters (Bang and Andrews 1974, Nelson 1985), but as with drogue tracking, tended to be used over small areas.

Although self-recording, moored current meters have been extensively employed - mainly on the shelf along the West Coast - these instruments have always been very expensive and hence never used in sufficiently large numbers to provide detailed information over substantial geographical areas. Also, for fear of damage by shipping, these instruments have rarely been deployed near the surface and thus have contributed mainly to our knowledge of subsurface flow patterns. Data obtained from current meter moorings have however revealed a larger degree of spatial and temporal variability of currents than previously known (Nelson 1985, Holden 1986, Holden 1987).

The contribution of satellite imagery to the study of surface advection patterns in South African waters.

The gradual introduction of satellite thermal band imagery into oceanographic studies during the 1970s and 1980s, brought about a distinct advance in our understanding of surface circulation patterns. Satellites provided the facility to track surface-drifting buoys over large distances and this made available a great deal of information on major current systems such as the Agulhas Current (eg. Gründlingh 1977, Gründlingh and Lutjeharms 1979) and the Benguela Current (Harris and Shannon 1979, Nelson and Hutchings 1983). However, more importantly, satellite infrared imagery made it possible, for the first time to 'see' the thermal structure of the ocean surface and through that to gain an insight into the complexity of the surface flow pattern (Fig.1). Infrared imagery was put to particularly good use in

unravelling the intricate Agulhas Current system (eg. Harris *et al.* 1978, Lutjeharms 1981).

The Agulhas Current is an ideal subject for study with satellite imagery since it is a very warm and well-defined western boundary current, clearly visible on most infrared images. The Current system also encompasses an enormous geographical area, parts of which are very far from land. For example, the interesting and important retroflection zone, where the Current abruptly reverses its flow direction, is located more than 400 km from land. The entire current system can be seen on a single cloud free image and this makes available information which would have been almost impossible to obtain from ship surveys. The advantage of the wide field-of-view afforded by the satellite sensors applies in a similar manner to many other situations in oceanographic research work. For example, at the bottom of Figure 1, a thin filament of cool water reaches about 300 km northwards from the cold Subtropical Convergence water to an eddy centred on 34°S : 14°E. The eddy is in this case located about the same distance from land as the Agulhas retroflection, yet seems to exert a direct influence on the nearshore marine environment, by drawing off a stream of cool water from the northwestern edge of the Agulhas Bank, directly south of the continent.

Satellite imagery such as Figure 1, has therefore made a substantial contribution to the present knowledge of current patterns, firstly through the fact that they cover large areas not easily studied by shipboard techniques, and secondly, by way

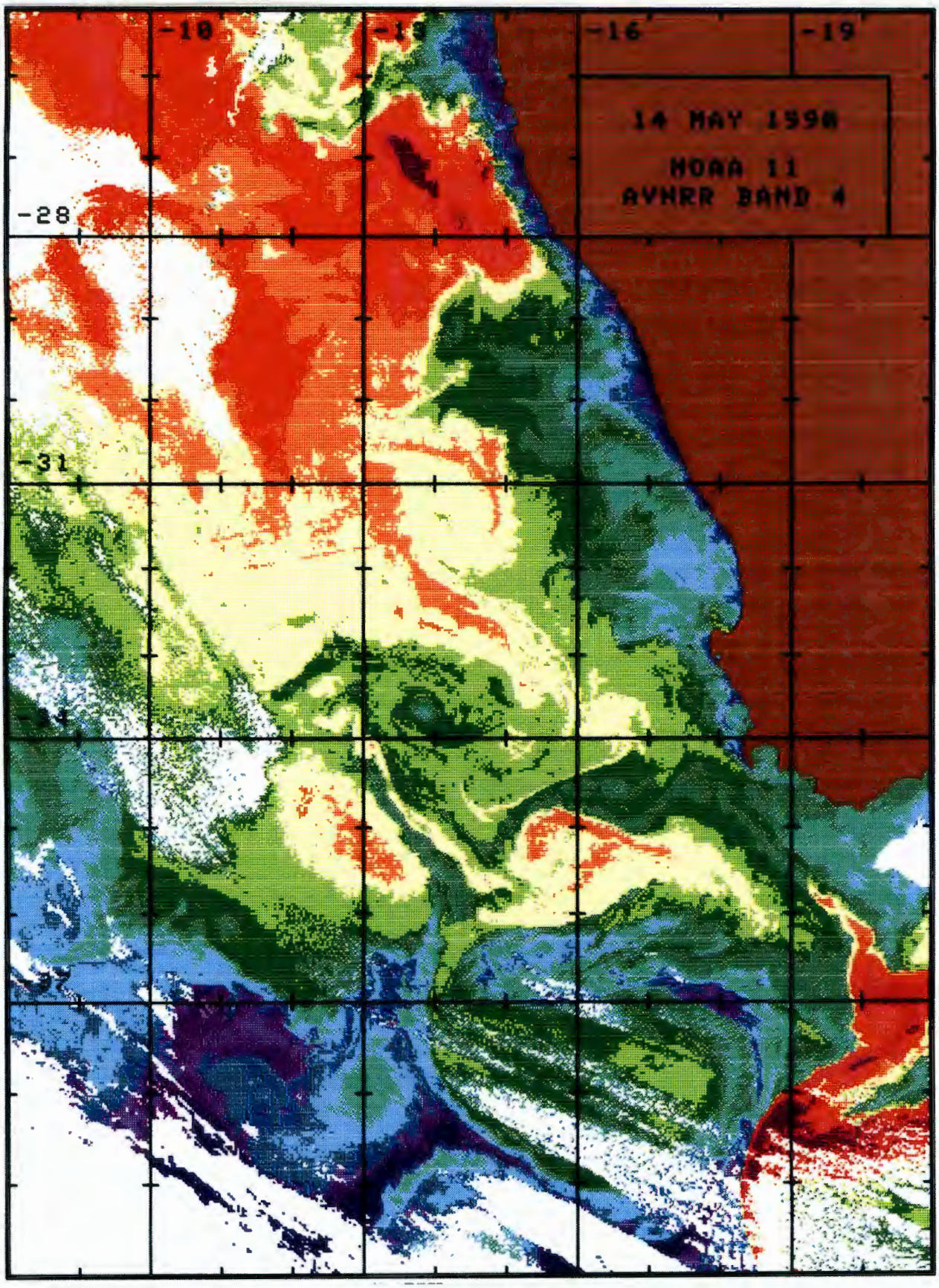


Fig.1 The sea surface temperature distribution pattern obtained from band 4 of the NOAA-11 Advanced Very High Resolution Radiometer. 14 May 1990.

of allowing certain conclusions regarding the flow field to be drawn from the sea surface temperature distribution. In the previous paragraph it was said that water was apparently being drawn off the Agulhas Bank by the eddy. This assertion was made in a kind of intuitive manner, based inter alia on the shape of the filament. In a similar vein one would judge from the thermal structure of the eddy that it is probably rotating in a cyclonic sense and from the warm water pattern in the bottom-right hand corner of Figure 1, that Agulhas Current water is being advected north-westward along the western edge of the Agulhas Bank (Fig. 4.1). It would however, seem to be completely feasible to perform a far more rational analysis of the flow pattern simply by comparing the location of a feature - for example the core of the cold eddy - with its location in a second image from a later date and from these positions compute a vector to indicate the speed and direction with which the feature was advected in the period between the two images. Such a process need not be restricted to macro-scale features such as the eddy, but could be applied to the small-scale structure visible in many parts of Figure 1. That way, a more detailed description of the flow field could be obtained. (Note : In this investigation only thermal imagery will be used. However other types of imagery, eg. visible band images, could be used in a similar manner).

Deriving advection velocities from series of satellite images. The concept of 'feature tracking'.

The process of tracing the progression of oceanographic features

is known as 'feature tracking' and forms the subject of the investigation described in this dissertation. The principle is straight forward and, as was seen above, is a logical extension of the observation of thermal structures on satellite imagery. In fact the basic idea was employed by Harris *et al.*(1978) in one of the very earliest applications of satellite imagery in South African oceanographic studies to estimate the phase velocity of wave-like perturbations on the Agulhas Current and also subsequently by Lutjeharms and van Ballegooyen (1988) to study the behaviour of the Agulhas retroflection.

In both these studies, feature tracking was restricted to following the motion of one or two large features over a long period of time using photographic reproductions of images. Tracking of small features had not been undertaken, due to the rigorous requirement for precise definition of feature positions imposed by such applications. The smaller the feature, the quicker is it destroyed or deformed beyond recognition by the turbulent mixing processes in the ocean. Therefore, in order to track small features, the time interval between the images needs to be small ie.less than one day. High resolution thermal imagery may be obtained from the operational NOAA series of satellites at intervals as small as three to four hours. However, since advection rates in the ocean are typically less than 50 cm/s, features can be expected to move only a few image resolution elements (pixels) over a period of hours, with the result that the derived flow pattern may easily be distorted by errors in the

geographical location of the features being tracked. The requirement for positional accuracy is usually satisfied by a high precision transformation of the pair of images to a common map projection.

Figure 1 has already been transformed to a suitable map projection and in principle, is in a suitable state for feature tracking, but it still lacks the spatial and radiometric (colour or grey-scale) resolution needed for the detection of small increments of motion. Photographic reproductions of imagery obtained from the satellite receiving station are ordinarily of much better quality but the problem of spatial and radiometric resolution remains. In fact, the required resolution can probably only be achieved through a computer video display system.

In 1984 the Sea Fisheries Research Institute (SFRI) acquired a small ARIES II image processing system which provided high definition display capability. The system has since been used primarily for the processing of NOAA AVHRR (Advanced Very High Resolution Radiometer) imagery data which are obtained in raw unprocessed form from the Satellite Applications Centre (SAC) of the Council for Scientific and Industrial Research (CSIR) in South Africa. The first attempt at computing advection velocities from imagery involved the use of both AVHRR and NIMBUS-7 CZCS (Coastal Zone Colour Scanner) imagery (Agenbag and Shannon 1988). In this attempt the images were not transformed to a common map projection but the positions of the features were computed individually using a set of algorithms published by Wilson *et*

al.(1981). Only a small number of advection vectors were computed on this occasion, but the experiment nevertheless demonstrated that it should be possible to derive synoptic velocity data over a larger geographical area and of greater density through this technique than is possible with any existing current measuring method. In making this statement one must emphasize the 'synoptic' and 'large geographical area' advantages of satellite imagery because ship-mounted Acoustic Doppler Current Profiling (ADCP) equipment which came into fairly common use during the late 1980s, is certainly capable of producing velocity data at high spatial densities, but spatial coverage and synopticity are limited by the relatively low speed of the ship.

Although the 1988 feature tracking experiment convincingly demonstrated to SFRI the potential value of the technique, it also showed clearly that a more efficient procedure was needed if the technique were to be exploited to its full potential. This realisation was underscored by the contents of a report which appeared in the literature at about that time in which Emery *et al.*(1986) described a method for automating the computation of surface advection velocities from satellite imagery and showed vector fields of much higher densities than achieved by SFRI. The stimulus provided by this paper, combined with the experience gained during the local experiment (Agenbag and Shannon 1988), gave rise to the research being reported here.

Structure of this thesis.

The primary objective of this thesis is to develop a feature tracking procedure suitable for the computation of sea surface advection velocities from two adjacent-in-time NOAA AVHRR images. As was indicated in the preceding introduction, this objective can readily be broken down into four, virtually independent, development steps, ie. development of navigation (geolocation) algorithms, creation of a routine for transforming the images to appropriate map projections, development of a feature tracking procedure and, finally, the evaluation of the overall process. In practice the navigation algorithms were used only as part of the transform procedure so that the descriptions of these two steps have been combined in chapter 2 of this thesis under the title 'Image registration'. The remaining steps are described in chapters 3 and 4, respectively.

Note that due to the disparate nature of chapters 2, 3 and 4 no comprehensive literature review is given at the beginning of the thesis. Instead relevant material will be discussed in the respective chapters.

The rationale behind the structure of the thesis is as follows:

In order to prepare a set of images for feature tracking they must be transformed to a common map projection; Mercator projections were used in this research. And the transformation process requires a means for relating image coordinates (line and

pixel numbers) to geographical coordinates (latitudes and longitudes). This function is provided by the navigation procedure. In Agenbag and Shannon (1988) this was based upon a set of algorithms which modelled the earth as spherical and the satellite's orbit as circular. Results obtained with this procedure were not satisfactory. Alternative procedures discussed in the literature were either of similar type or required ground based satellite tracking data not available to SFRI. The major part of chapter 2 is therefore devoted to the development of an improved navigation procedure for AVHRR imagery using readily available satellite orbital parameters. The method developed here is based on an elliptical satellite orbit and an ellipsoidal earth model and, while only slightly more complex than the initial procedure used, proved capable of considerably improved navigational accuracy. The remainder of Chapter 2 is devoted to a description and testing of a procedure for transforming the imagery to a Mercator projection, making use of the new navigation routine.

The automated procedure for computing advection velocities described by Emery *et al.*(1986) was thoroughly investigated (Chapter 3). The method was implemented in different forms, but results obtained during testing were generally unsatisfactory. This prompted the adoption of a 'semi-automated' feature tracking concept which essentially involved incorporation of the template matching technique, used by the automated procedure, into the framework of a manual tracking process.

In Chapter 4 three case studies are presented in which the transformation and semi-automated feature tracking procedures were used to derive advection patterns in the South East Atlantic. Although the case studies described are of much oceanographic interest in themselves, due to the degree of detail available for the description of offshore features, the primary objective was to evaluate the performance of the procedures.

Chapters 5 and 6 summarise the main results obtained in the thesis along with a few concluding remarks. Thereafter the listings of five computer programs developed during this investigation follow in a series of appendices.

2. IMAGE REGISTRATION.

The automated feature tracking procedure to be described in chapter 3, in brief, strives to match a small section of one image with a similar size section of a second image. If, for example, the section of the first image is centred on scan line number n and pixel number m, then the neighbourhood of line n, pixel m in the second image is tested for a match. Assuming the best match is achieved with the image section centred on line n+i, pixel m+i, then the surface advection vector at point (n, m) is taken to be described by the line connecting point (n, m) with (n+i, m+i). Needless to say, the image coordinates (n, m) and (n+i, m+i) should represent the same geographical coordinates (latitude and longitude) in each of the two images. This condition may be achieved by either registering one of the images to the other, or by registering both images to a map projection.

The tools for performing image-to-image registration are standard elements of any image processing system, including the ARIES II system used by the Sea Fisheries Research Institute. The procedure involves two steps viz.:

(a) Calculation of the transform function which transfers the x', y'-coordinates of the first image (called the "slave") to the x, y-coordinates of the second image (called the

"master"). The function used, consists of a pair of 3rd order polynomials of the form:

$$x' = a_o + a_1x + a_2y + a_3x^2 + a_4xy + a_5y^2 + a_6x^3 + a_7x^2y + a_8xy^2 + a_9y^3$$

$$y' = b_o + b_1x + b_2y + b_3x^2 + b_4xy + b_5y^2 + b_6x^3 + b_7x^2y + b_8xy^2 + b_9y^3$$

If 10 pairs of line/pixel coordinates [(x_i, y_i) and (x_i', y_i'), $i=1,10$] are known, the coefficients, a_i, b_i $i=0,9$, can be determined. These coordinates are referred to as "ground reference points" or "ground control points" and are the coordinates of geographical features such as islands, capes or large water reservoirs which can be readily identified on both images. A minimum of 10 points are needed to solve for the coefficients, but usually more are used. To avoid biasing the transforms the reference points should be evenly distributed over the area to be transformed.

(b) The second step involves interpolation of the original pixel values to yield the pixel values for the transformed image. For each pixel in the transformed image - coordinates (x, y) - the line/pixel coordinates (x', y') in the original image can be derived from the transform equations; these will normally be of fractional form and the value at that point needs to be determined from the neighbouring pixels. The ARIES II software use a "cubic convolution" procedure which calculates the value as a weighted mean of the neighbourhood.

Although less labour is involved by simply registering one image to the other i.e. to regard one as the "slave" and the other as the "master", the vectors derived from such a scheme would only be defined in image coordinates and would need to pass through a second computation stage to establish them in absolute units of knots and degrees relative to north, as would be desired. It was therefore decided to transform both images to a Mercator projection. However, irrespective of the mode of transform, ground reference points are required for setting up the transform equations and since such points are not obtainable over the sea, the requirement for an even spread of points cannot be met, so that the integrity of the transforms - and eventually the translation vectors - would become more and more questionable with increasing distance from land. The way out of this problem is to make use of a navigation routine which calculates the geographical coordinates for a given picture element from a mathematical model of the satellite orbit and scanner characteristics.

2.1 IMAGE NAVIGATION

Earth observation satellites are launched into orbits with very small eccentricities (ca 0,001 in the case of NOAA), hence the tendency is to model these orbits as circles for ease of computation. Several descriptions of algorithms for image navigation appeared in the literature e.g. Legeckis and Pritchard

(1976), Wilson *et al.* (1981) and Ho and Asem (1986) - these were all based on circular orbit and spherical earth models. Although relatively simple and fast in terms of computation time, the basic assumptions of constant altitude and angular velocity associated with this type of model, is expected to lead to navigational errors.

In the case of an elliptical orbit with an eccentricity of 0,00162 and semi-major axis of 7229,8 km (NOAA 9, 86/7/27) the geocentric distance to the satellite varies between 7218,1 km and 7241,5 km. The earth radius varies between 6378,1 km at the equator and 6365,6 km at latitude 50°. By using averages for these quantities an uncertainty of about 17,9 km in altitude is introduced. Altitude appears in the trigonometric calculation of the geocentric angle between the ground point (the sub-satellite point i.e. the point on the globe directly below the satellite) and a pixel within a particular scanline. Inaccuracies in altitude thus lead to displacement of the ground point from its true position and hence to errors in computation of the line and pixel numbers for a given geographical position. The effect becomes more pronounced with increasing distance from the centre of the scan. Using again the example of a NOAA 9, north bound track image of 86/7/27, for which the inclination was 99,0°, the orbital period 102,08 minutes and the satellite at 30°S, we can calculate that an altitude error of 18 km would result in a pixel error of about 6 at a point 200 pixels from the centre; at 800 pixels from the centre the error becomes equivalent to 11 pixels.

In the case of a circular orbit model, with orbit parameters as above, the travel time from latitude 50°S to the equator could be calculated as 865,3 seconds. In the case of an elliptical orbit model, angular velocity varies with position in the orbit, being largest at perigee. Depending upon the position of perigee, the travel time may be calculated to range between 862,57 and 868,0 seconds. In other words, the circular orbit model may over or under estimate the travel time by about 2,7 seconds. In practice the use of reference points will reduce this effect since we are only really concerned with changes in angular velocity over the distance of the image area, in which case the maximum expected deviation is about 0,47 seconds per 10° of latitude - this is equivalent to an error of 3 scan lines.

Although the conditions described are more or less representative of the extreme situation, the errors - and the pixel error in particular - are far too large to be acceptable for the determination of ocean currents through feature tracking. Ho and Asem (1985) were able to reduce navigation errors by computing altitude from ground reference data, made possible through having access to precise data on the time and longitude of the satellite's equator crossing. Such data are not available to us. Therefore after initial experimentation with the algorithms of Wilson *et al.* (1981) and Ho and Asem (op. cit.) a model was adopted based on an elliptical orbit and ellipsoidal earth. Although this model still retained elements of the circular orbit model in the sense that spherical trigonometry is used for many calculations, it does allow both altitude and angular velocity

to vary with distance along the track and generally produces more accurate navigation results than were obtainable with the first algorithms. In the following sections the model is described in some detail and its performance discussed.

2.2 THE ELLIPTICAL ORBIT/ELLIPSOIDAL EARTH MODEL

The model is the basis used for developing procedures by which

(a) the geographical coordinates(latitude and longitude) are computed for a given pair of image coordinates (line and pixel), and

(b) the image coordinates are computed for given geographical coordinates.

In each case the essential step is to determine the satellite's altitude and the geographical coordinates of the point directly below it, at the moment a particular scan line was being sampled. This position is referred to as the nadir point, the sub-satellite point, or simply, the ground point. The term "ground-point" will be used here. Once the ground-point coordinates are known, pixel coordinates are obtained through straightforward geometry. The mathematical descriptions which follow are therefore structured so as to provide the equations necessary for the computer algorithms which will perform the navigational

calculations. The formulae are generalised to be applicable to

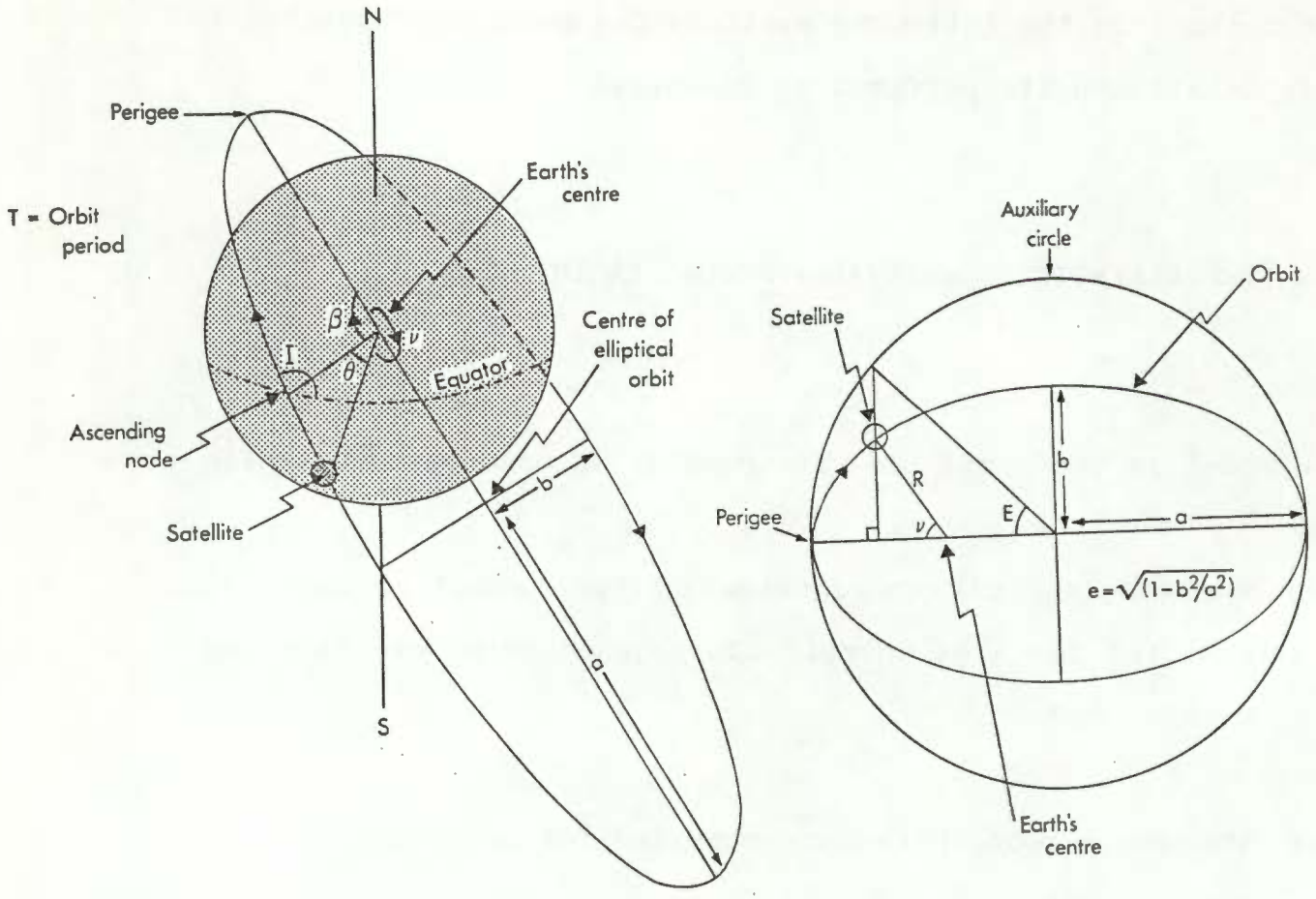


Fig. 2.1: The geometrical configuration for the orbits of NOAA polar orbiting satellites.

images in all hemispheres as well as to ascending and descending tracks. To achieve generalisation the following notation is adopted:

(i) Southern latitudes and eastern longitudes are negative.

(ii) The trackline inclination is positive for ascending tracks and negative for descending tracks.

(iii) The generalisation factor, G, is introduced, where G=1 for ascending tracks and G=-1 for descending tracks.

2.2.1 Mathematical description of orbit dynamics

The configuration for the elliptical orbit of NOAA satellites is illustrated in Figure 2.1. The symbols used are :

a,b = The semi-major and semi-minor axis.

ß = Argument of perigee. The geocentric angle, measured in the orbit plane from the ascending node to perigee. Positive in flight direction.

ν = True anomaly. The geocentric angle measured in the orbit plane from perigee to the satellite's ground point.

θ = The acute geocentric angle measured in the orbit plane from the ascending or descending node to the satellite's ground point. The angle is positive in the flight direction e.g for an ascending track, $-90° < \theta < 0°$ in the southern hemisphere and $0° < \theta < 90°$, in the northern hemisphere.

I = Inclination of the orbit. Measured westward from the equatorial plane at the ascending node. It is taken to

be positive for ascending tracks and negative for descending tracks.

E = The eccentric anomaly. The angle at the centre of the ellipse (orbit) measured in the orbit plane from perigee to the projection of the satellite's position onto a circle circumscribing the ellipse. Positive in flight direction.

R = The radial distance from the earth's centre to the satellite.

e = The eccentricity of the elliptical orbit.

The satellite ephemeris parameters defining the orbit, i.e. semi-major axis, inclination, period and eccentricity are assumed to be known. They are not constants but vary slowly with time. The argument of perigee, for example, move at a rate of 2,83 per day in the counter rotation direction. These data may be obtained from the 'APT Predict (TBUS) bulletins', transmitted daily by KWBC Washington,DC on the Global Telecommunication Service network and is in South Africa provided by the S.A.Weather Bureau. In practice, this Institute acquires the data once per week and a linear interpolation, with time, is carried out to compute a single set of data for a time corresponding to the centre of the image.

Calculation of angle θ and the trackline inclination at the ground point coordinates latitude, ϕ_1, longitude, λ_1

If we assume the satellite's nadir track to be a great circle, θ may be calculated by solving the spherical triangle ABC, in Figure 2.2

$$\sin(I_2)/\sin(\phi_1) = \sin(90°)/\sin(\theta)$$

$$\therefore \sin(\theta) = \sin(\phi_1)/\cos(I_o) \qquad (\cos(I_o)=\sin(I_2))$$

$$\therefore \theta = \text{arc sin}\,(\sin(\phi_1)/\cos(I_o)) \qquad (2.1)$$

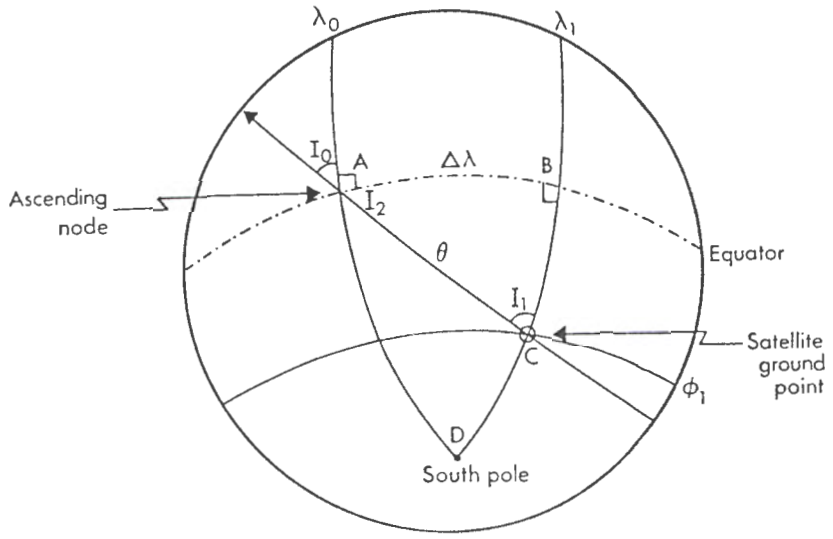


Fig. 2.2 Equation diagram. See text.

C = the satellite's ground point, A = the nodal point,
D = the south pole,
BC = the geocentric angle, ϕ_1
AC = the geocentric angle, θ
I = the trackline inclination (west of north) at the equator
i.e. $I_o = I - 90°$

Equation (2.1) applies to both hemispheres since ϕ_1 is positive in the northern hemisphere and negative in the south. However, for application to descending tracks θ, must be multiplied by -1. The general equation is then given by:

$$\theta = G \times \arcsin\left(\sin(\phi_1)/\cos(I_0)\right) \quad , \qquad (2.1)$$

where G = +1 for ascending tracks and G = -1 for descending tracks.

θ is measured from the equator to the ground point and will thus be positive in the northern hemisphere and negative in the southern hemisphere for an ascending track. The inverse applies in the case of a descending track.

The trackline inclination, I_1, at the ground point, is calculated from spherical triangle ACD in Figure 2.2:

$$\sin(180-I_1)/\sin(90) = \sin(I_0)/\sin(90-\phi_1)$$

$$\therefore I_1 = \text{arc sin}\left\{\sin(I_0)/\cos(\phi_1)\right\} \quad , \qquad (2.2)$$

Calculation of the true anomaly, ν, for angle θ

For an ascending track ν may be defined by inspection of Figure 2.1 as:

$$\nu = 360 - \beta + \theta$$

It can similarly be shown that for a descending track

$$\nu = 180 - \beta + \theta$$

These equations are good for both hemispheres and may be generalised for ascending/descending tracks as:

$$\nu = 270 + 90 \times G - \beta + \theta \quad , \qquad (2.3)$$

where if $\nu < 0$ then $\nu = \nu + 360$

and if $\nu > 360$ then $\nu = \nu - 360$

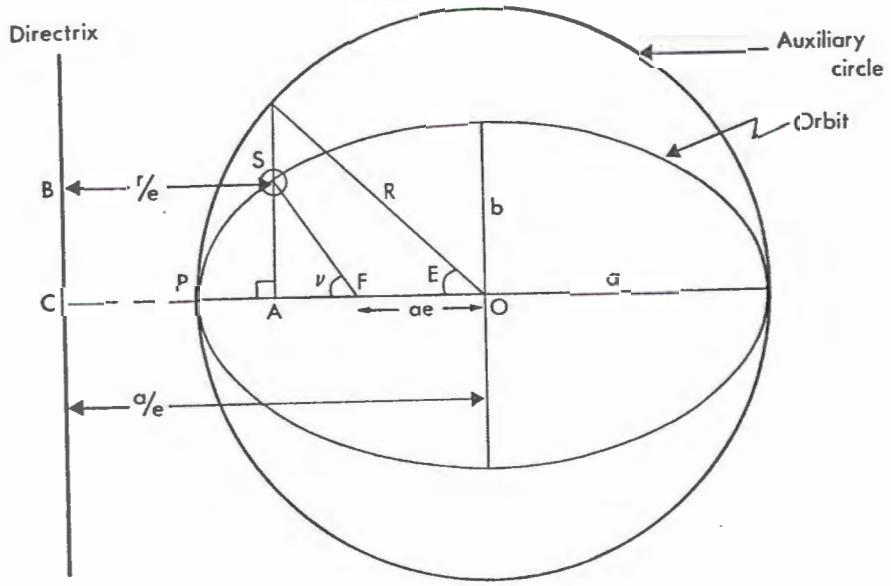


Fig. 2.3 Equation diagram. Refer to text.

Calculation of the distance R from the earth's centre to the satellite for true anomaly **ν**

With the earth's centre at the focus of the ellipse (F in Fig 2.3) and the satellite at S, an expression for the distance R may be derived from the geometric identities for the ellipse and the conic property SB = R/e

$$\therefore \quad R = e \times SB \qquad (e = \text{eccentricity})$$

$$= e(FC - FA)$$

$$= e((a/e - ae) - R.\cos(\nu))$$

$$= a(1-e^2) - e.R.\cos(\nu)$$

$$\therefore \quad R(1+e.\cos(\nu)) = a(1-e^2)$$

$$\therefore \quad R = a(1-e^2)/(1+e.\cos(\nu)) \quad , \qquad (2.4)$$

This equation applies to all conditions.

Calculation of the eccentric anomaly E, from the true anomaly **ν** and the geocentric distance R.

From Figure 2.3 we get:

$$\cos(E) = AO/a$$

$$= (ae+R.\cos(\nu))/a$$

$$\therefore \quad E = \text{arc } \cos(e+R.\cos(\nu)/a) \quad , \qquad (2.5)$$

If $\nu > 180$ then $E = 360 - E$

Equation 2.5 applies to all conditions.

Another expression for E which will be used is

$$E = \arccos\left(\frac{e+\cos(\nu)}{1+e.\cos(\nu)}\right) \quad , \qquad (2.5b)$$

(Escobal 1976, p.118)

Calculation of the travel time, t, from the equator to the ground point.

The time t_0 is the time in seconds during which the satellite travels the acute angle ν_0 (or corresponding angle E_0) from perigee to the equator. This time is given by Kepler's equation as:

$$t_0 = (T/2\pi)(E_0 - e.\sin(E_0)) \quad , \qquad (2.6)$$

Where T = the nodal period(sec)

e = eccentricity

E_0 = eccentric anomaly calculated from:

i) $\theta=0$ in eq.(2.3) i.e. $\nu_0=270+90G-\beta$

ii) ν_0 in eq.(2.4) i.e. $R_0=a(1-e^2)/(1+e.\cos(\nu_0))$

iii) R_0 and ν_0 in eq.(2.5) i.e. $E_0=\arccos(e+R_0.\cos(\nu_0)/a)$

The time t_1 is the time the satellite travels from perigee to the ground point and is given by eq.(2.6) as:

$$t_1 = (T/2\pi)(E_1 - e.\sin(E_1)) \quad , \qquad (2.7)$$

where the eccentric anomaly E_1 is calculated for the ground point latitude ϕ_1 through equations (2.1, 2.3, 2.4 and 2.5)

The travel time t, from the equator to the ground point is then

simply given by:

$$t = t_1 - t_0 \quad , \tag{2.8}$$

To force t to carry the same sign as θ, i.e. to adhere to the adopted notation for θ and t to be positive in the flight direction as measured from the relevant nodal point to the ground

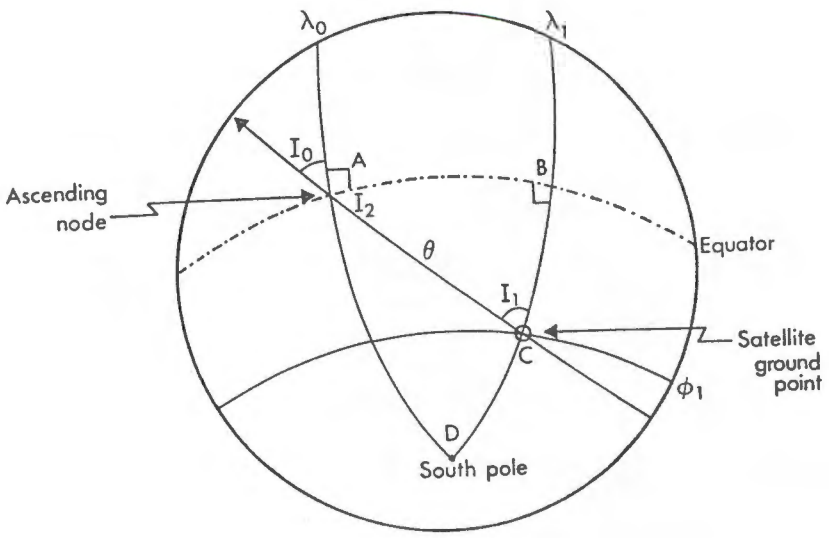


Fig. 2.4 Equation diagram. Refer to text.

I_o = the trackline inclination at the equator (I_o=I-90)
I_1 = the trackline inclination at ground point latitude ϕ_1

point, the nodal period must be either added to or subtracted from the result of equation (2.8) when t and θ are of opposite sign:

If $\theta>0$ then t=t+T and

if $\theta<0$ then t=t-T

Calculate nodal longitude λ_0 for satellite ground point latitude ϕ_1 and longitude λ_1

In the spherical triangle ABC (Fig. 2.4) line segment AB represents the geocentric angle between the ground point longitude and the ascending node longitude.

From the sin formula:

$$\sin(\Delta\lambda) = \sin(I_1).\sin(\phi_1)/\sin(90-I_0)$$
$$= \sin(I_1).\sin(\phi_1)/\cos(I_0)$$

From equation (2.2) $\sin(I_1) = \sin(I_0)/\cos(\phi_1)$

$$\therefore \sin(\Delta\lambda) = \sin(I_0).\sin(\phi_1)/\cos(I_0).\cos(\phi_1)$$

$$\therefore \Delta\lambda = \arcsin\{\tan(I_0).\tan(\phi_1)\} \quad , \qquad (2.9)$$

Equation (2.9) is correct for all hemispheres and tracks, provided the notation for inclination is adhered to - ie. I and I_0 must be negative for descending tracks.

If the earth was not rotating and the orbit static, then λ_0 would have been given by

$$\lambda_0 = \lambda_1 - \Delta\lambda$$

However, since the earth rotates at the rate of 360° per 86164,09 seconds (= one sidereal day), the nodal longitude must be adjusted by

$$\omega = 360/86164{,}09 \text{ degrees per sec.} \qquad (2.10)$$

In addition the orbit plane is not fixed but precesses slowly due

to the oblateness of the earth and results in the nodal longitude being displaced at the rate of Ω deg./day.

The rate of precession can be approximated by:

$$\Omega = -1,5 \times 0,00108263 \times r^2 \times \sqrt{\mu} \times R^{3,5} \times \cos(I) \qquad (2.11)$$

(Duck and King 1983)

where r = earth mean radius (6371 km)

R = geocentric distance to the satellite.

μ = earth's gravitational constant ($3,98601 \times 10^5$ km^3/sec^2)

I = orbit inclination at the equator (deg. westward from the equator)

The precession rate is small compared with ω hence R may be replaced by the semi-major axis, a, without significant loss of accuracy.

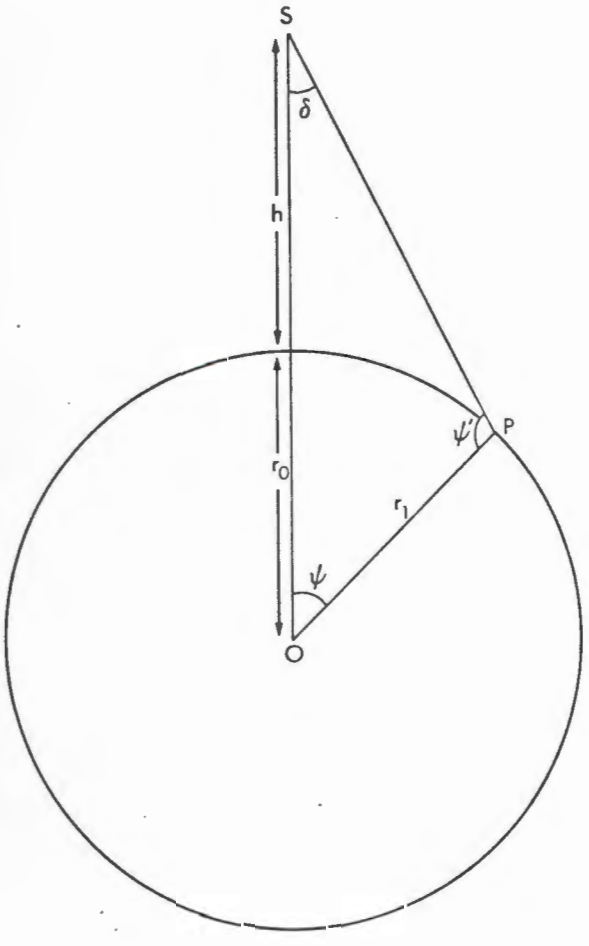
The combined rotation/precession rate is given by $\omega-\Omega$ and when t is the travel time from the equator to the ground point, as calculated with eq. (2.8). Then the nodal longitude is:

$$\lambda_0 = \lambda_1 - \Delta\lambda - t(\omega-\Omega) \quad , \qquad (2.12)$$

Equation (2.12) is made good for all hemispheres and tracks through use of the adopted notations for I and t.

2.2.2 Mathematical description of the off-nadir viewing geometry

Section 2.2.1 dealt with the dynamics of the satellite and assumed, as a starting point for most calculations, that the geographical coordinates of the ground point were known. In this section the satellite is considered to be stationary while executing a particular scan line and the geometry is described for calculating the latitude and longitude of a pixel within the scan, from the ground-point coordinates and also the inverse, ie. ground-point coordinates from pixel coordinates.



S = satellite
P = pixel
O = earth's centre
δ = scan angle
h = satellite's altitude
r_0 = earth's radius at the ground-point
r_1 = earth's radius at the pixel

Fig. 2.5 Geometry for computation of the satellite's altitude.

Satellite attitude

Wilson *et al.* (1981) in their navigation routine for Nimbus-7 CZCS imagery, took attitude variations (roll, pitch and yaw) into consideration. Legeckis and Pritchard (1976), working with VHRR imagery from NOAA-4 and -5, reported attitude variations of less than 0,5° and applied a correction for roll deviation obtained by observing the positions of the horizon on their images. Ho and Asem (1986) considered it unnecessary to correct AVHRR images for attitude variation and Brush (1985) reported attitude variations for NOAA satellites to be less than 0,1° and regarded errors from this source as negligible. On grounds of the last two assertions no provision was made for attitude corrections in this model.

The scan angle δ

This is the angle measured at the scanner, and in the vertical plane, between the nadir and pixel p.

Digital NOAA AVHRR imagery provided by the Satellite Applications Centre (SAC) of the CSIR, normally contain 2048 pixels per scan. In terms of image coordinates (lines/pixels), the nadir position is thus defined by pixel number 1024,5. Each pixel represents an incremental scanner rotation angle of 0,95 milliradians (Kidwell, 1983). Therefore the scan angle δ for pixel p is given by:

$$\delta = (p-1024{,}5) \times 0{,}054431 \text{ degrees} , \qquad (2.13)$$

As shown in Figure 2.5, scan angle δ is subtended by angle ψ at the earth's centre. This angle is important for subsequent

calculations. In plane triangle SPO:

$$\psi' = 180 - (\delta + \psi)$$

$$\therefore \sin(\psi') = \sin(\delta + \psi)$$

Further, from the sin formula

$$\sin(\psi')/(h+r_0) = \sin(\delta)/r_1$$

$$\therefore \sin(\psi') = \{(h+r_0)/r_1\} \times \sin(\delta)$$

$$\therefore \sin(\psi+\delta) = \{(h+r_0)/r_1\} \times \sin(\delta)$$

$$\therefore \psi = \arcsin[\{(h+r_0)/r_1\}.\sin(\delta)] - \delta \quad , \qquad (2.14)$$

In equation (2.14) the term $(h+r_0)$ is equal to R, the radial distance from the earth's centre to the satellite as may be calculated for the ground-point latitude by equation (2.4). The radius, r_1 at the pixel can be obtained from:

$$r_1 = r_e(1 - f.\sin^2(\phi_2)) \quad \text{(Brush 1985)} \quad , \qquad (2.15)$$

where r_e = earth's equatorial radius (6378,137 km)

ϕ_2 = pixel latitude

and $f = (r_e - r_p)/r_e = 0,00335$

r_p = polar radius of the earth (6356,752 km)

Scan skew

The scan direction is perpendicular to the direction of flight but because of the satellite's motion during the scan, the actual trace over the earth's surface is slightly skewed relative to the perpendicular. So that, if the satellite is considered stationary while executing the scan, the relative bearings for the start and end points of the scan - when measured clockwise from flight

direction - are about 90,01° and 270,01° respectively.

The AVHRR scanner rotates clockwise - when looking in the flight direction - and therefore, on descending tracks in the southern hemisphere, pixel number 1 is the most westerly and also the first in time, line number 1 being the most northerly. On ascending tracks the first pixel , in time, is to the east of the track and line one is the most southerly. However, in the digital imagery supplied by the SAC, CSIR, the data are repositioned so that the first record is the northernmost and the first pixel count in the record corresponds with the most westerly sample. This should be borne in mind since it affects the relative bearing of pixels.

The bearing of pixel number 2048, relative to the flight direction is given by γ in the expression:

$$\gamma = \arccos(\mu/\psi_m) \quad \text{(Ho and Asem 1986)} \quad , \quad (2.16)$$

where μ = the arc distance travelled by the satellite during the time of half a scan line.

ψ_m = the geocentric distance corresponding to half a scan.

For half a scan, the scan angle δ' is given by equation (2.13):

$$\delta' = (2048 - 1024{,}5) \times 0{,}054431$$

and ψ_m is obtained with δ' in (2.14). The sensor completes six scans per second, ie. it rotates through 6 x 360° per second and hence will take $\delta'/6x360$ seconds for half a scan. Therefore

$$\mu = (\delta'/(6x360)) \times (360/T)$$

Since the scan skew is very small it is sufficiently accurate to use the mean angular velocity of 360/T in this case. The skew angle γ varies with latitude through the presence of R and r_1 in the equation for ψ_m (equation 2.14). In practice γ is calculated only once per image area - again due to the scan skew being small.

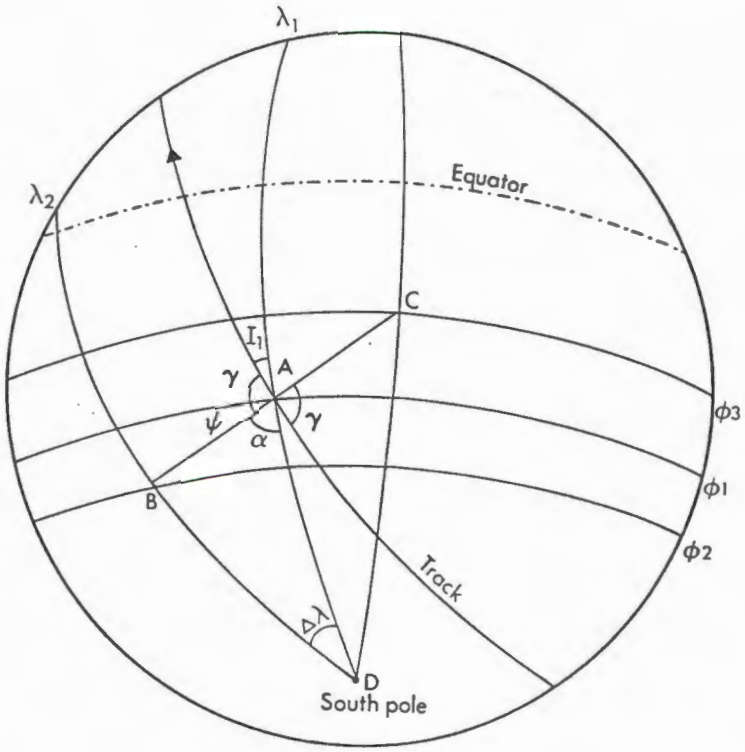


Fig. 2.6 Sensor viewing geometry.

Calculation of pixel coordinates (ϕ_2, λ_2) from ground point coordinates (ϕ_1, λ_1)

The viewing geometry for a scan line with the satellite at point A on an ascending track in the southern hemisphere is illustrated

by Figure 2.6. In this figure the satellite track and the scan line, BAC, are both considered to be great circles. I_1 is the trackline inclination west of north, at ground-point latitude ϕ_1, ψ the geocentric angle between the ground point and the pixel and γ the scan-line relative bearing.

The pixel latitude ϕ_2, may be calculated from the spherical triangle, ABD:

$$\cos(90-\phi_2) = \cos(\psi).\cos(90-\phi_1)+\sin(\psi).\sin(90-\phi_1).\cos(\alpha)$$

$$\therefore \sin(\phi_2) = \cos(\psi).\sin(\phi_1)+\sin(\psi).\cos(\phi_1).\cos(\alpha)$$

$$\text{and } \phi_2 = \text{arc } \sin(\cos(\psi).\sin(\phi_1)+\sin(\psi).\cos(\phi_1).\cos(\alpha) \qquad (2.17)$$

$$\text{where } \alpha = 180 - (\gamma + I_1) \qquad (2.18)$$

If the cosine formula is applied to triangle ACD (Fig. 2.6) - i.e. for pixel numbers larger than 1024,5 - then angle α will be defined as $\gamma+I_1$ and $\cos(\gamma+I_1) = -\cos(180-(\gamma+I_1))$. The equation for γ (equation 2.16) yields the same result (about 89,99°) for both parts of the scan; angle ψ however changes sign from negative, for pixels less than 1024,5, to positive for the other half. Application of equation (2.17) will, however, yield the correct result provided γ and α are defined as by (2.16) and (2.18). The expression is also correct for the northern hemisphere, as well as for descending tracks - as long as I_1 is in this case taken to be negative as per the adopted notation.

Pixel longitude λ_2, is obtained by application of the sin formula

to spherical triangle ABD (Fig. 2.6):

$$\sin(\Delta\lambda)/\sin(\psi) = \sin(\alpha)/\sin(90-\phi_2)$$

$$\therefore \quad \Delta\lambda = \arcsin(\sin(\alpha).\sin(\psi)/\cos(\phi_2)) \qquad (2.19)$$

and

$$\lambda_2 = \lambda_1 - \Delta\lambda \qquad (2.20)$$

Equations (2.19) and (2.20) are correct for all conditions.

A practical problem arises in the application of equation (2.14) for calculation of angle ψ, since the radius r_1 is a function of the unknown latitude ϕ_2. An iterative approach is therefore used whereby r_1 is initially calculated with the ground-point latitude ϕ_1 and then recalculated twice with the resulting ϕ_2. The radius does not vary rapidly with latitude, making more iteration steps unnecessary.

Calculation of ground-point coordinates (ϕ_1, λ_1) from pixel coordinates (ϕ_2, λ_2)

This is a more complex computation since not only is ψ a function of the unknown latitude ϕ_1, but also angle α, via its dependence on the trackline inclination, I_1 (eq. 2.6).

It is actually possible to eliminate α from the trigonometric equations as was shown by Wilson *et al.* (1981) by substitution of

$$\alpha = (180-\gamma) - \arcsin(\sin(I_0)/\cos(\phi_1))$$

(equations 2.18 and 2.2)

into equation 2.17 which after re-arrangement yields:

$$\sin(\phi_1) = (a.b/(b^2+c^2)) \pm (c/(b^2+c^2)) \times \sqrt{(d(b^2+c^2)-a^2)} \qquad (2.21)$$

where $a = \sin(\phi_2) - \sin(\psi).\sin(\gamma).\sin(I_0)$

$b = \cos(\psi)$

$c = -\sin(\psi).\cos(\gamma)$

$d = 1 - \sin^2(I_0)$

The second term in equation (2.21) is small (< 0,000029) and will affect the value of ϕ_1 by less than 0,005° for a maximum scan angle at $\phi_1 = 70°$. An error of 0,005° in latitude is equivalent to 0,55 km, or about half nominal nadir pixel size. At lower latitudes and smaller scan angles the effect is less and this term may consequently be discarded with little loss of accuracy, so that:

$$\phi_1 = \text{arc } \sin(a.b/(b^2+c^2)) \qquad (2.22)$$

Although α is now eliminated an iteration is still required since ψ in (2.21) is a function of ϕ_1. Therefore, ϕ_1 and λ_1 were solved as follows:

Initialise the iteration by computing ψ and α for pixel latitude ϕ_2 (equations 2.14 and 2.18). Use these to get $\Delta\lambda$ (equation 2.19) and finally insert ψ, α and $\Delta\lambda$ into the trigonometric relationship:

$$\tan(a)/\tan(x) = \cos(b)/\cos(c) \qquad (2.23)$$

where $a = \{(90-\phi_2)+\psi\}/2$

$b = (\alpha-\Delta\lambda)/2$

$c = (\alpha+\Delta\lambda)/2$

$x = (90-\phi_1)/2$

so that

$$\phi_1 = 90 - 2.\text{arc tan}\{\tan(a).\cos(c)/\cos(b)\} \qquad (2.24)$$

The process is repeated twice with the estimated ϕ_1, after which the ground point is known to a precision of better than a quarter pixel size.

2.2.3 Algorithm for nodal longitude and travel time

As mentioned in section 2.3, most orbit parameters are obtained by interpolation from the TBUS-bulletin data. The GMT time for each scan line appears within the auxiliary data which precede the image data of each record on the data tapes supplied by the SAC. The time, for a scan line in the centre of the relevant image section, is used for a linear interpolation between the data from two consecutive TBUS-bulletins. In this manner the following variables are obtained:

a = semi-major axis (km)

T = nodal period (minutes)

I = inclination of the orbit plane (degrees westward from the equator. Taken as positive for ascending- and negative for descending tracks)

e = eccentricity

ß = argument of perigee (deg.)

Two more parameters are required to complete the orbit

description, i.e. the longitude of equator crossing (nodal longitude) and the time of crossing. These may also be obtained from the TBUS-bulletins but, because of interpolation errors combined with inaccuracies in the model, such data proved to be unsatisfactory. Use was therefore made of ground reference points, i.e. positions for which both image coordinates (lines/pixels) and geographical coordinates (latitude/longitude) are known, to calculate the nodal longitude and time. This procedure possibly leads to loss of absolute accuracy in nodal longitude and time, but since they are computed through the same model used for subsequent image navigation procedures, navigation accuracy is improved. In fact, the nodal time is not used at all but rather the satellite's travel time from the equator to the reference point standardised to an arbitrary point (scan line number 1) in order to use the average from several reference points rather than one point.

The algorithm for computing nodal longitude λ_0, and standard travel time t_s, from a reference point with line l_2, pixel p_2, latitude ϕ_2, and longitude λ_2, is as follows:

i) Convert the reference point geodetic latitude ϕ_2' to a geocentric latitude ϕ_2:

$$\phi_2 = \arctan(c.\tan(\phi_2')) \qquad (2.25)$$

where $c = (r_p/r_e)^2 = 0,9933$

r_p and r_e are the polar and equatorial radii of the earth.

ii) Calculate δ from (2.13)

iii) Calculate r_1 from (2.15)

Set ground point latitude ϕ_1 to pixel latitude ϕ_2 and

iv) Calculate θ, I_1, ν and R with (2.1, 2.2, 2.3 and 2.4)

v) Calculate ψ with (2.14) and α with (2.18)

vi) Calculate $\Delta\lambda$ with (2.19) and a new ground point latitude ϕ_1, with (2.24)

vii) Test ϕ_1 against the previous value of ϕ_1 and return to step (iv) for another iteration if the error limit is exceeded. The iteration converges to better than a quarter pixel accuracy within three cycles.

viii) Calculate ground point longitude λ_1, with (2.20)

ix) With ν and R from the last iteration cycle, calculate E_1 with (2.5) and t_1 with (2.7)

x) With θ=0 calculate ν_0 with (2.3), R_0 with (2.4), E_0 with (2.5) and t_0 with (2.6)

xi) The travel time t, from the equator to the ground point is then obtained from (2.8)

xii) Calculate $\Delta\lambda$ with (2.9) and λ_0 with (2.12)

xiii) The standard travel time from the equator to scan line number one is then given by:

$t_s = t + G(l_2 - 1)/6$ (G=1 for ascending tracks, G=-1 for descending tracks)

It should be noted that t_0, as computed in step (x) above, is a constant for a given argument of perigee and can therefore be computed once and stored for later use.

This algorithm is contained within subroutine NOAN1 of the

program NOANAV (Appendix A)

2.2.4 Algorithm for the calculation of latitude ϕ_2, and longitude λ_2, for a pixel specified by line number l_2 and pixel p_2

For this computation it is assumed that the nodal longitude λ_0, and the standard travel time t_s, have been determined.

i) Calculate the travel time t, from the equator to scan line l_2 from

$$t = t_s - G(l_2-1)/6$$

ii) Calculate t_1, the travel time from perigee to the satellite's ground point with (2.8). If $t_1<0$ then $t_1=t_1+T$ and if $t_1>t$ then $t_1=t_1-T$

iii) Calculate E with (2.7). This is done through the Newton solution for the general equation

$AX + B.\sin(WX) + C = 0$ and

$X_{i+1} = X_i - (AX_i + B.\sin(WX_i) + C)/(A + BW.\cos(WX_i))$

In the case of equation (2.7), A=1, B=-e, W=1 and

$C=-2\pi.t_1/T$

$\therefore E_{i+1} = E_i-(E_i-e.\sin(E_i)-2\pi t_1/T)/(1-e.\cos(E_i))$

The iteration is initiated with $E_1=2\pi t_1/T$ and converges within 5

iteration cycles.

iv) From (2.5b) ν is computed as

ν = arc cos[(cos(E)-e)/(1-e.cos(E_1)]

if E>180 then ν=360-ν

v) Calculate θ with (2.3). If θ>180 then θ=θ-360 and if θ<-180 then θ=θ+360

vi) Calculate the ground point latitude ϕ_1, with (2.1)

vii) Calculate $\Delta\lambda$ with (2.9) and the ground point longitude, λ_1 from time *t*, derived in step (i), inserted into equation (2.12)

viii) Use ν from step (iv) to compute R with equation (2.4)

ix) Calculate I_1 with (2.2), α with (2.18) and δ with (2.13). Set pixel latitude ϕ_2, initially to ground point latitude ϕ_1

x) Compute r_1 with (2.15) and ψ with (2.14)

xi) Compute pixel latitude ϕ_2 with (2.17). Test ϕ_2 against the previous value of ϕ_2, if the difference exceeds the error limits then return to step (x). Three iteration cycles are sufficient.

xii) Use ψ from the final iteration cycle to compute $\Delta\lambda$ with (2.19) and finally the pixel longitude λ_2 with (2.20)

xiii) Convert the pixel geocentric latitude to geodetic latitude using (2.25)

This algorithm is contained within subroutine NOAN2 of program NOANAV (Appendix A).

2.2.5 Algorithm for the calculation of line and pixel numbers for a given latitude and longitude (ϕ_2, λ_2)

As in section 2.2.4 it is assumed that the nodal longitude λ_0, and the travel time from the equator to scan line number one t_s, have been obtained and also that the time t_0, (travel time from perigee to equator) has been saved.

The algorithm consists, in broad terms, of an iteration procedure similar to that used by Wilson *et al.* (1981). As a first guess the pixel is taken to be in the centre of the image, the ground point latitude and longitude are calculated and from those the travel time and nodal longitude. This nodal longitude is compared with λ_0 and the pixel estimate adjusted to the left or right depending on the sign of the error. In detail the procedure is as follows:

i) Convert the pixel geodetic latitude to geocentric using

equation (2.25).

ii) Calculate r_1 with (2.15)

iii) Set left pixel limit p_l=0 and the right pixel limit p_r=2049.

iv) Estimate pixel number $p=(p_l+p_r)/2$ and calculate δ with equation (2.13).

Set ground point latitude ϕ_1, initially equal to the pixel latitude ϕ_2.

v) Calculate θ with (2.1), I_1 with (2.2), ν with (2.3), R with (2.4) and E with (2.5)

vi) Calculate ψ with (2.14) and α with (2.18).

vii) Calculate $\Delta\lambda$ with (2.19) and the ground point latitude ϕ_1, with (2.24). Repeat steps (v) to (vii) two more times.

viii) Use $\Delta\lambda$ from the final iteration step (vii) in equation (2.20) to get the ground point longitude λ_1.

ix) Insert E from the final iteration cycle, step (iv), into (2.7) to obtain t_1 and then calculate t with (2.8) - using the previously saved value for t_0.

x) Calculate $\Delta\lambda$ with (2.9) and λ_0' with (2.12).

xi) Compare λ_0' with λ_0 and if $|\lambda_0'-\lambda_0|$ >error limit then adjust the pixel limits and return to step (iv) for the next iteration cycle. If $\lambda_0'>\lambda_0$ then $p_r=p$ and if $\lambda_0'<\lambda_0$ then $p_l=p$.

xii) The required pixel number is then p, as assigned in step (iii) during the final iteration cycle. The line number is calculated from t, derived in step (ix) during the final cycle and

$$l = G \times 6(t_s - t) + 1$$ (Were G=1 for ascending- and G=-1 for descending tracks)

This algorithm is contained within subroutine NOAN3 of program NOANAV (Appendix A).

2.3 ALGORITHM TEST RESULTS

Considerable pains were taken in the design of the procedures, described in the preceding sections, to ensure they could be used with all AVHRR imagery irrespective of the flight direction or position on the globe. However, all available imagery were obtained from the CSIR's receiving station at Hartebeesthoek and only cover the southern and eastern hemispheres around South Africa, consequently the algorithms could not be properly tested

for the northern and western hemispheres. Artificial data were used to check the algorithms' behaviour in these parts and these checks suggest the routines function correctly but these tests are not regarded as conclusive and no results will be presented.

With regard to testing the procedures we are primarily interested in the model's behaviour away from the ground reference points,used for calculation of nodal longitude, λ_0, and travel time, t_s. Systematic increases in error size in both the scan and track directions are to be expected and will determine the usable range of the algorithms in terms of acceptable limits of navigation error. We further need to investigate the consistency of the algorithms with different sets of orbit parameters, particularly variations in perigee position. And finally,since the accuracy of the procedure depends to a very large degree on the accuracy of ground reference points,precision of reference point identification needs consideration. Though last mentioned, the matter of reference points needs to be addressed first since all other tests will be carried out through use of reference points.

2.3.1 Precision in the identification of ground reference points.

A ground reference point - as previously stated - is a terrestrial feature which can be identified on the image and a cartographic map of the same area. Both image coordinates (line/ pixel) and geographic coordinates(latitude/longitude) are

therefore defined for such a point. Coastal features such as islands, peninsulas and capes are suitable reference points on AVHRR imagery but do not provide a sufficiently broad data base as required by the envisaged algorithm testing. Coordinates were therefore also extracted for 145 dams, on the principle that these could be expected to be clearly visible on infra-red imagery - a necessary condition since NOAA descending tracks over South Africa are almost invariably during the night. To avoid problems with varying water levels in the dams, reference points were taken either close to the retaining walls or in the case of the larger dams, at branch points or other identifiable positions less dependent on the actual water level.

Latitudes and longitudes were recorded to a precision of 10^{-4} degrees with an uncertainty interval of ±0,0005°. Image coordinates were generally recorded as a fraction of a line or pixel. This practice, which may seem incongruous, arise in a situation such as where an island appears in the image as an even number of pixels of uniform grey level; the centre of the island should then logically, be on the boundary between two pixels e.g. 100,5. Should the radiance from pixel 101 be less than pixel 100 then this is taken to imply that a larger portion of 101 overlapped the surrounding water and hence the centre is regarded as being closer to pixel number 100 é.g. 100,2. There are obvious weaknesses in this stratagem so that image coordinate precision is judged to be no better than 0,5 lines/pixels. Inaccuracies in ground reference data, however, stem mostly from the need to be able to identify, on the image, the very same point for which

the geographical coordinates have been extracted, and errors in judgement in this respect, could easily lead to the image coordinates being wrong by one, two, or possibly, more lines or pixels. It is desirable to obtain a measure of this uncertainty and to what extent it changes with distance from the image centre.

Table 2.1: Orbit parameters for the NOAA 9 ascending track image on 27 July 1986

Semi-major axis	=	7229,80371 km
Nodal period	=	102,07997 min
Inclination (westward from equator)	=	99,00047 deg
Eccentricity	=	0,00162
Argument of perigee	=	40,64991 deg
AVHRR scan rate (seconds per scan)	=	0,16667 sec
AVHRR digitisation step angle	=	0,05443 deg

By good fortune an excellent test image was available from the NOAA 9 ascending pass on 27 July 1986 (see Table 2.1 for orbit parameters). On this occasion the satellite's track passed approximately through the centre of the South African sub-continent and almost the entire land region from 23°S southwards was cloud free. Being a day time pass, reference points were selected using enhanced AVHRR bands 1 and 4 data. A total of 38 coastal and 126 other (dams) reference points could

be identified. These ranged from line 13 to 1332 and pixels 407 to 1952; latitudes 22,4 to 34,8°S and longitudes 14,4 to 33°E. The average nodal longitude from these points was 14,5883°E, with a standard deviation of 0,0090° and the average travel time from the equator to line 1 was -386,9185 sec, with a standard deviation of 0,1305 sec.

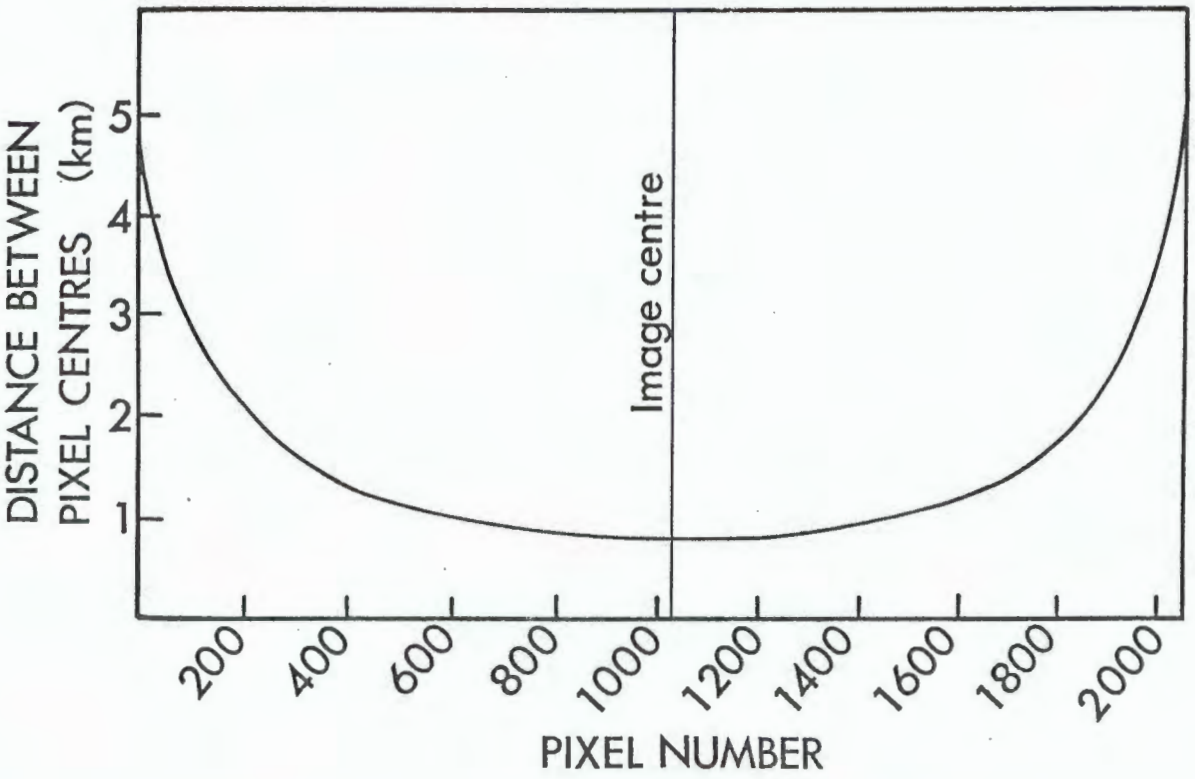


Figure 2.7 Variation of AVHRR pixel size with pixel number. For satellite ground point at 25°S

An error in the reference point line number has little effect on the nodal longitude and affects the travel time by a constant 0,1667 sec per line. Pixel error, on the other hand, affects

mainly the nodal longitude but since pixel size increases towards the sides of the image (Fig 2.7), the effect is not constant, ranging from approximately 0,01° near the centre to about 0,017° at pixel number 1800. However, taking 0,1667 and 0,01 as averages, the standard deviations indicate a scatter of 0,78 lines and 0,90 pixels in the reference points - which should of course in theory all compute to identical nodal longitudes and travel times. Actually, when inserting the average nodal longitude and travel time into the algorithms and doing a backwards calculation of the line and pixel numbers for the 164 reference points, the root-mean-square(RMS) line and pixel errors are 0,78 lines and 0,67 pixels respectively.

In reality, neither of the two results above provide much insight into reference point precision since the data set of reference points span a large range of lines and pixels so that the resulting ranges of nodal longitudes and travel times are likely to reflect the behaviour of the model rather than errors in reference point selection. To overcome this problem, eleven subsets, of five or more, reference points were selected so that each subset represents a cluster with the overall range of lines and pixels not exceeding 100 (arbitrary). Table 2.2 provides an example of one such cluster and Table 2.3 lists the pertinent results from all sets.

In Figure 2.8 the standard deviations in λ_0 and t_s, from the clusters, are presented as functions of the cluster distance from the centre of the scan. The increase in pixel size towards the

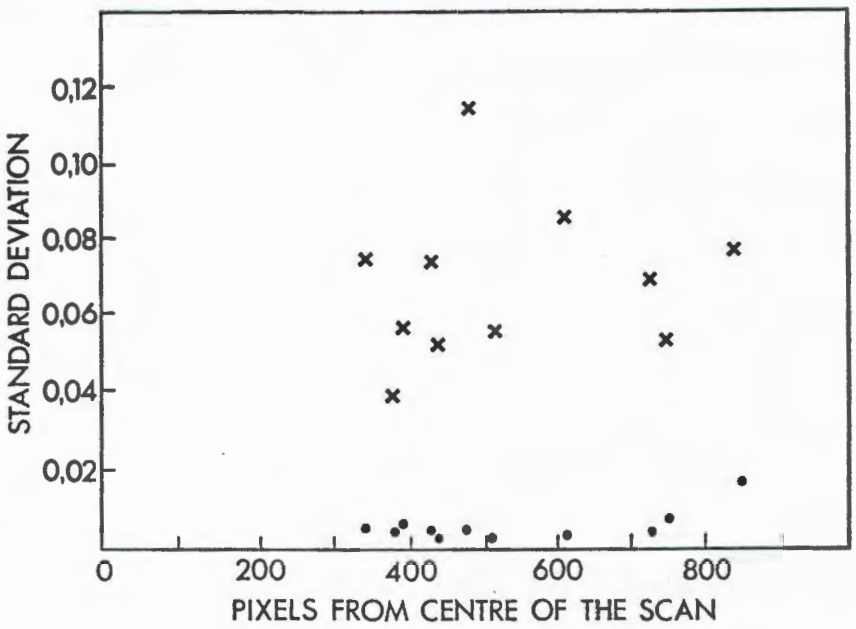


Figure 2.8 Standard deviation of λ_0 (.) and t_s (x) for clusters of reference points.

edge of the image (Fig. 2.7), coupled with the larger optical path length and associated atmospheric attenuation, is expected to have an adverse effect on the precision of reference point identification nearer the edges. In Figure 2.8, the deviation in λ_0 is fairly uniform and small, except for the last two points which show an upward trend - possibly on account of the edge effect.

The deviation of t_s (Fig. 2.8) is about an order of magnitude larger than that of λ_0 and shows no definite trend. When the standard deviations of λ_0 and t_s are expressed in terms of pixels

Table 2.2: A cluster of reference points from the NOAA 9 image of 27 July 1986. λ_0 is the nodal longitude and t_s the travel time from the equator to image line one.

Line	Pixel	Latitude(°S)	Longitude(°E)	λ_0(°E)	t_s(sec)
1147,0	696,0	33,1088	19,7801	14,5802	368,7822
1160,6	619,0	33,3460	19,0436	14,5816	368,8440
1165,4	619,0	33,3898	19,0511	14,5781	368,7960
1170,0	700,0	33,3284	19,8823	14,5852	368,7723
1199,5	643,7	33,6870	19,3936	14,5825	368,7810
1203,0	647,0	33,7163	19,4271	14,5737	368,7761
1208,0	645,0	33,7742	19,4261	14,5790	368,8917
1209,0	612,0	33,8252	19,0947	14,5879	368,8449
1233,0	624,0	34,0388	19,2665	14,5758	368,7891
1235,5	624,0	34,0655	19,2755	14,5780	368,8310
Mean	1193,1	643,0		14,5802	368,8108
σ				0,0043	0,0399

Cluster no. 1 ; N = 10 ; lines 1147-1235,5; pixels 612-700

Table 2.3: Results from 11 clusters of reference points. N = the number of points in each subset ; λ_0 and σ_λ are the mean and standard deviation of the nodal longitude and E_p the standard deviation expressed in terms of pixels. t_s and σ_t are the equivalent parameters for travel time with E_l the standard deviation as expressed in lines.

No	N	Average line	Average pixel	λ_0	σ_λ	E_p	t_s	σ_t	E_l
1	10	1193,1	643,0	14,5802	,0043	,41	368,8108	0,0399	0,24
2	11	1217,9	596,9	14,5794	,0045	,41	368,7899	0,0705	0,42
3	6	449,2	413,9	14,5878	,0039	,30	368,9761	0,0856	0,51
4	5	1249,8	1501,6	14,6013	,0053	,45	368,7889	0,1151	0,69
5	8	859,6	1754,6	14,5952	,0041	,24	368,9576	0,0692	0,41
6	10	913,6	1532,4	14,5888	,0025	,21	368,9502	0,0560	0,34
7	7	994,8	1413,8	14,5910	,0060	,56	368,8809	0,0575	0,34
8	5	904,1	1365,1	14,5903	,0047	,47	368,9251	0,0746	0,45
9	6	873,2	1460,6	14,5904	,0031	,29	369,0019	0,0526	0,32
10	6	666,5	1869,2	14,5759	,0172	,75	369,0748	0,0777	0,47
11	6	591,5	1774,8	14,5854	,0078	,44	369,0456	0,0530	0,32

and lines (E_p and E_l in Table 2.2) the trend for λ_0 is removed, as well as the size difference between σ_λ and σ_t. For both parameters, nine of the clusters then exhibit standard deviations of less than half a pixel or line, which is actually a little better than anticipated. It should also be pointed out that outliers had not been removed from the cluster data in Table 2.2, as would be done in practice. For example, removal of one point from cluster number ten will reduce σ_λ from 0,0172 to 0,0055.

Conclusions :

i) Reference point selection is as precise as may reasonably be expected. Standard deviations in the image coordinates are generally better than half a scan line or pixel.

ii) Spurious reference points occur which will degrade accuracy of λ_0 and t_s. Several reference points should be used and outliers discarded.

iii) There is some evidence that reference point precision decreases beyond ±725 pixels from the centre of the scan. For computation of λ_0 and t_s, reference points should thus be selected between pixel number 300 and 1750.

iv) Under these circumstances σ_λ is expected to be better than 0,006° and σ_t better than 0,115 sec for a randomly selected cluster of points. Outliers must be removed to reduce σ_λ to

less or equal 0,005° and σ_t to 0,08 sec.

2.3.2 Algorithm behaviour in flight direction.

Four sets of reference points could be found within the data base of 164 reference points for the image of 27 July 1976, which gave reasonably large ranges of line numbers but within a fairly narrow pixel range. The idea being that these lines of points would provide insight into the algorithms' behaviour in the satellite track direction with possible interference from along-scan variations removed through enforcement of a narrow pixel range within each group. The standard travel time t_s, and nodal longitude λ_0, for the largest of the four sets are illustrated in Figures 2.9a and b as functions of line number. Table 2.4 lists results from all four sets.

The graph of t_s (Fig. 2.9a) shows a slight downward trend with increasing line numbers but the slope is small (-0,0003 sec/line after deletion of outliers), so that t_s is expected to change by 0,1667 sec (the equivalent of one scan line) over a distance of 555 scan lines. The largest slope was obtained with set number 3 (Table 2.3), i.e. -0,00039, which corresponds with a one line error over 427 scans. In other words if t_s is computed from a reference point near the middle of a 500 scan line image, the systematic line error at the top and bottom of the image, should be ±0,5 lines which is of the same magnitude as the uncertainty in the reference point as was shown in 2.3.1

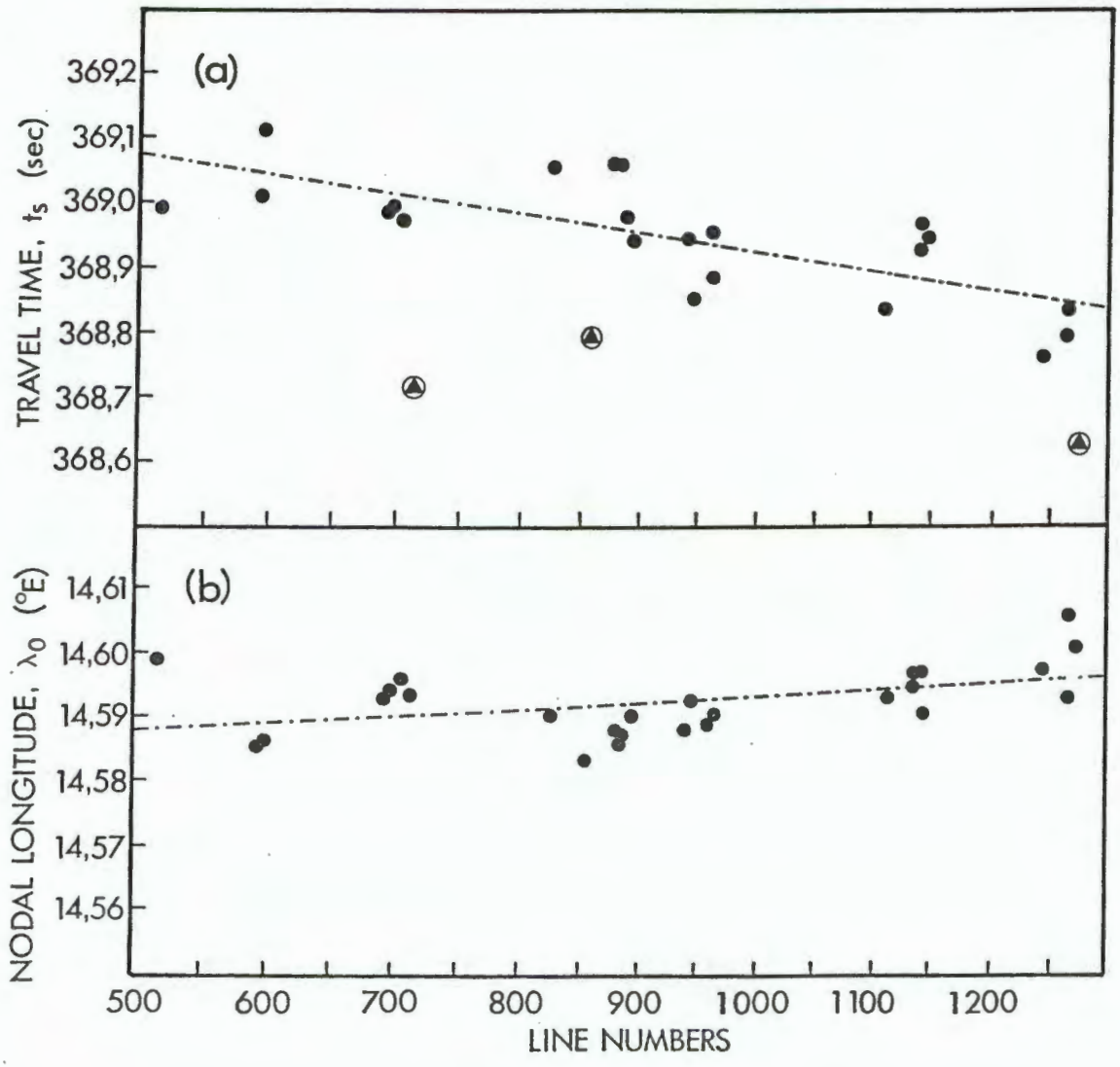


Fig. 2.9: Variation of t_s and λ_0 with scan line number. Subset number 4 of reference points in flight direction.

Since the nodal longitude is computed from t_s, the graph of λ_0 (Fig 2.9b) exhibits an eastward drift with increasing line numbers, corresponding to the progressive underestimation of travel time. The largest slope, 0,0000193 deg/line, was therefore - as in the case of t_s - given by data set number 3. The largest systematic error predicted for λ_0 over 500 scan lines would be

about 0,0096° which is equivalent to less than one pixel.

Table 2.4: Results from subsets of reference points in the satellite flight direction. NOAA 9 ascending track on 27 July 1986.

No	N	Line range	Pixel range	λ_0 slope	λ_0 intercept	t_s slope	t_s intercept
1	14	13-577	407-437	,0000025	14,5847	-,00011	368,9953
2	12	757-1259	517-579	,0000028	14,5767	-,00032	369,1571
3	9	727-1322	1337-1414	,0000193	14,5722	-,00039	369,2562
4	25	516-1272	1482-1549	,0000107	14,5827	-,00030	369,2262

N = number of points in the subset
λ_0 = nodal longitude (°E)
t_s = travel time from equator to scan line one (seconds)

In a test computation one reference point from near the middle of data set number 4 (scan line 893) was used to compute λ_0 and t_s. These were then used to compute the line and pixel numbers for the remaining members of set 4 (3 outliers excluded as before). The resulting RMS line and pixel errors were 0,50 and 0,46 respectively. And if the latitudes and longitudes are similarly computed then the RMS latitude error is 0,0047° and the RMS longitude error is 0,0048°.

Conclusions :

The tests performed indicate that the model is not completely stable in the flight direction, resulting in navigation errors equivalent to about one line and one pixel over a range of 400 to 500 scan lines. Since the variation with distance is near

linear, nodal longitude and travel time should be computed from reference points near the centre of the image or, if possible, spread evenly over the image range, in which case the systematic errors over 400-500 scan lines are expected to be reduced to about half a line/pixel.

2.3.3 Algorithm behaviour in the scan direction.

This was investigated in a similar fashion as the along-track behaviour i.e. by selecting subsets of reference points from the data base for the image of 27 July 1986, such that the pixel number range is large but the line number range in each set, small. The results from section 2.3.2 indicated that the model was reasonably stable in the track direction, hence the restriction on the line number range could be relaxed to 200 lines. Even so, only two sets of points could be assembled on account of the lack of dams (and thus reference points) in the arid western- and central parts of the country. The first, and best set - in terms of data distribution - encompassed the pixel range 527 to 1 638 (Figs 2.10a, b), while the second set had a larger pixel range, i.e. 516 to 1952, but with a very uneven distribution of points (Fig. 2.11a, b).

The graphs of travel time t_s against pixel number (Figs 2.10a and 2.11a) show in each case a tendency for t_s to increase with

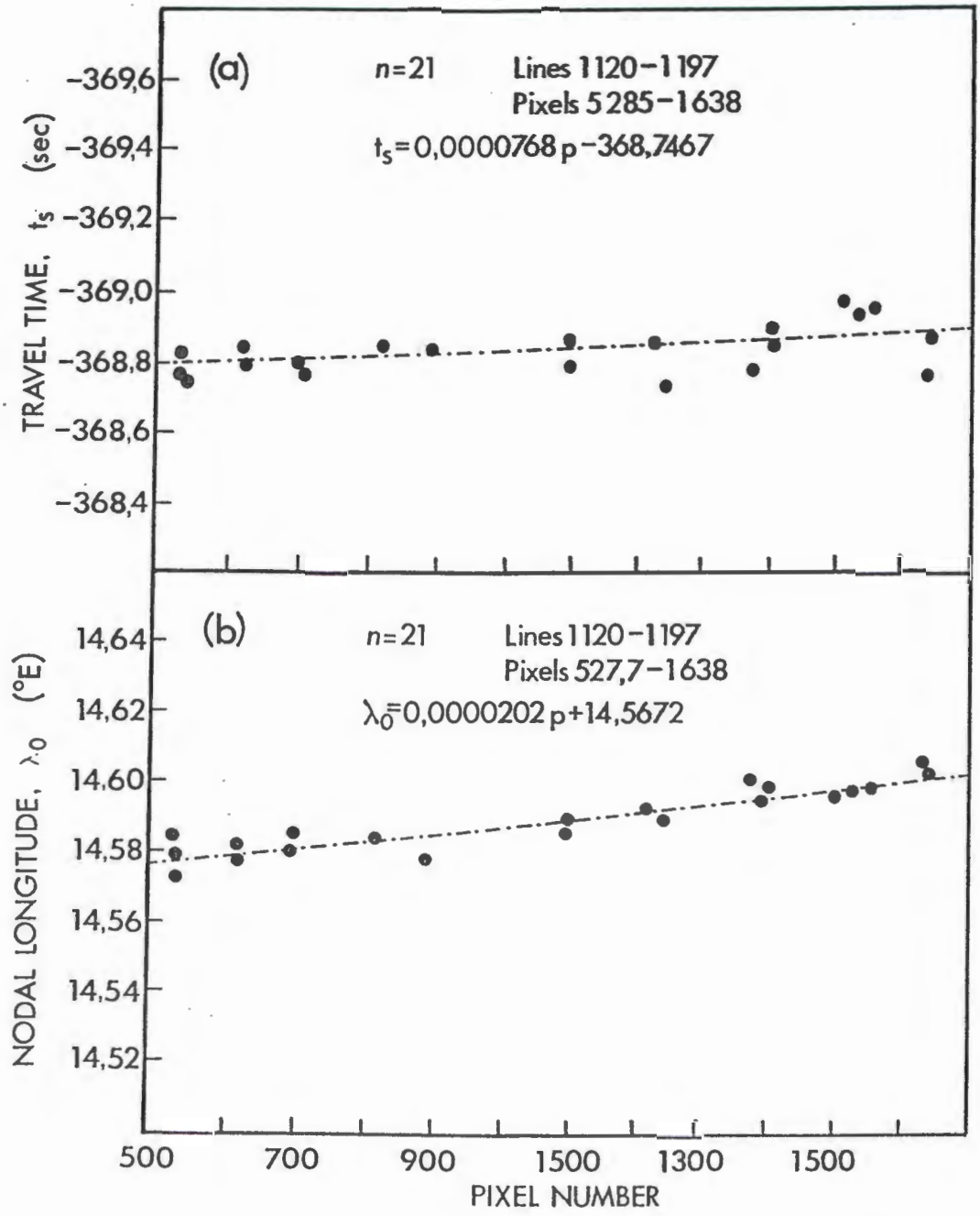


Fig. 2.10: Variation of t_s and λ_0 with pixel number. Subset number one of reference points in the scan direction.

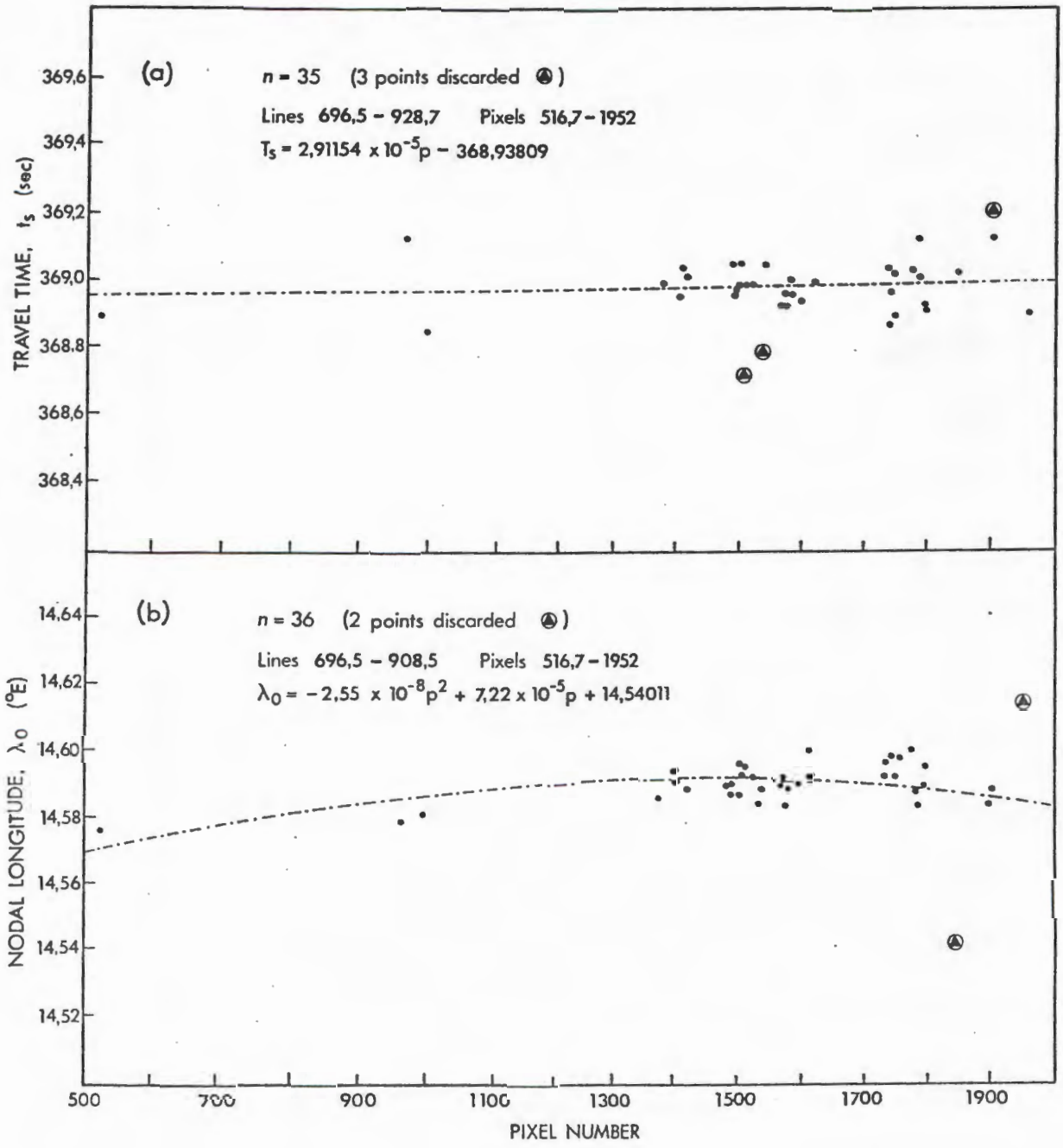


Fig. 2.11: Variation of t_s and λ_0 with pixel number. Subset number two of reference points in the scan direction.

pixel number. The linear slopes are however small, $2,9 \times 10^{-5}$ and $7,7 \times 10^{-5}$. seconds/pixel respectively, and represent a maximum systematic line error of less than one scan line over the entire width of the image.

The graphs of nodal longitude versus pixel number (Figs 2.10b and 2.11b) exhibit linear slopes of similar magnitude to t_s. Figure 2.11b shows signs of non-linearity and could be best fitted with a quadratic function. The non-linearity however, seems to be introduced mainly by the few data points with pixel numbers larger than about 1750. If these points are excluded, a linear fit with a slope of $7,2 \times 10^{-6}$ and intercept 14,5781, is obtained. The quadratic and linear regressions for Figure 2.11b suggest systematic deviations in nodal longitude of 0,0063° and 0,0073° respectively, from the centre of an AVHRR image to its edge. This is equivalent to better than half a pixel. The slope for Figure 2.10b is larger, i.e. $2,02 \times 10^{-5}$, and indicate a systematic error of 0,021° or 0,87 pixels, from the centre to the edge.

In an inverse calculation, two points near pixel 1100 in the first set and four points near pixel number 1400 in the second set were used to compute nodal longitudes and travel time. These were then used to compute firstly, the line and pixel numbers and secondly, the latitudes and longitudes for the remaining points in each set - i.e. 19 and 29 points respectively (5 outliers removed from the second set). This gave RMS line errors of 0,49 and 0,45 and RMS pixel errors of 0,81 and 0,52 respectively. The

RMS latitude errors were 0,0039° and 0,0043° and the RMS longitude errors 0,0099° and 0,0061°.

Conclusions :

The tests suggest that for λ_0 and t_s computed from reference points near the centre of the scan line, the image coordinates of a position to the west of the ascending track, will be under estimated by up to about one pixel and half a scan line, and on the other side of the track, over estimated by the same amount. This asymmetry is typical of an under estimate of the satellite's altitude. Altitude manifests itself on λ_0 and t_s through ψ which defines the geocentric angle between the pixel and the satellite nadir position. From equation 2.14 it is seen that ψ varies with $\{(h + r_0)/r_1\}.\sin(\delta)$. So that the influence of an error in altitude (h) will:

(a) increase non-linearly with increasing scan angle (δ)

(b) be more pronounced at higher latitudes where r_1 is smaller

(c) be more pronounced in that half of the scan where the satellite "looks" towards the pole i.e. for an ascending track image in the southern hemisphere this will be the western half of the scan.

Similar effects would however arise through errors in the scan angle (δ) such as may be introduced by attitude variations. If, for instance, δ were to be adjusted by an angle of -0,02° - such

as would arise from a roll deviation - the curvature in Figure 2.11b is all but removed.

The deviations in λ_0 and t_o are however not excessive and rather similar to those found to occur in the flight direction (section 2.3.2), only in this case the larger error is in the pixel or longitude coordinates. Consequently, if image size is restricted to a width of 500 pixels, the error is reduced to the equivalent of about half a pixel, which was the order of precision found for reference point definition in section 2.3.1.

Table 2.5: Orbital parameters for the NOAA 9 descending track image at 01:04 GMT on 28 July 1986

Semi-major axis	=	7 229,804 km
Nodal period	=	102,080 min
Inclination (westward from the equator)	=	-99,0005°
(The negative sign indicates a descending track)		
Eccentricity	=	0,00162
Argument of perigee	=	39,34352°
AVHRR scan rate (seconds per scan)	=	0,16667 sec
AVHRR digitisation step angle	=	0,05443°

2.3.4 Algorithm behaviour in the case of a descending track image

In the preceding sections the NOAA 9 ascending track image of 27 July 1986 was used to test the geolocation algorithms. Approximately 12 hours later, at 01h00 on 28 July, NOAA 9 was on a descending track more-or-less through the centre of the South African sub-continent and a good cloud free image was obtained

(see Table 2.4 for orbit parameters). Although this image covered a smaller section of land than the previous, a total of 39 coastal and 79 other reference points could be identified. These ranged from scan line 12 to 796 and pixels 320 to 1842; latitudes 26,1 to 34,7°S and longitude 14,9 to 31,1°E. The average nodal longitude from these points was 28,6877°E with a standard deviation of 0,0119° and the average travel time from the equator to scan line 1 was 466,1113 seconds with a standard deviation of 0,1379 seconds.

Since visible band imagery could not be employed, reference points had to be identified by means of the infrared data only. This proved to be fairly difficult due to low contrast between water in the dams and the land. To obtain maximum radiometric resolution the work was done on the raw AVHRR bands 3 and 4 images.

The difficulty with reference point identification is reflected by the standard deviations in λ_0 and t_σ for 7 clusters of points (Table 2.6). When compared with Table 2.2 for the day time image, these deviations are now somewhat larger, representing average line and pixel uncertainties (E_l and E_p) of 0,60 lines and 0,57 pixels compared with 0,41 and 0,41 for the day time image.

Removing outliers :

Considering the greater uncertainty in reference point identification with this type of image, a means of removing outlying

points was needed. A simple iterative procedure yielded acceptable results. The procedure consists of:

a) An attempt to reduce σ_λ and σ_t to a value less or equal to the deviation in λ_0 and t_s, which would result from half a pixel and half a scan-line error in the reference-point

Table 2.6 Results from seven clusters of reference points.

#	N	Average line	Average pixel	λ_0	$\sigma\lambda$	E_p	t_s	σ_t	E_l
1	9	87,0	342,2	28,6753	,0053	,34	465,9335	,1014	,61
2	12	709,9	916,3	28,6874	,0060	,64	466,0891	,0743	,45
3	6	300,4	1496,3	28,6924	,0049	,43	466,1534	,1367	,82
4	8	60,7	1736,2	28,6731	,0055	,34	466,2601	,0812	,49
5	9	133,7	1548,7	28,6920	,0097	,81	466,2100	,1178	,71
6	8	166,6	1560,9	28,6916	,0102	,84	466,2338	,1041	,62
7	16	732,1	889,1	28,6868	,0058	,62	466,0594	,0831	,50

Effect of removing outliers from the clusters:

Cluster #	Number of points removed	Before removing outliers		After removing outliers	
		E_p	E_l	E_p	E_l
1	1	0,34	0,61	0,34	0,48
2	1	0,64	0,45	0,52	0,40
3	3	0,43	0,82	0,10	0,39
4	0	0,34	0,49	0,34	0,49
5	4	0,81	0,71	0,28	0,44
6	2	0,84	0,62	0,27	0,45
7	4	0,62	0,50	0,46	0,33
Average		0,57	0,60	0,33	0,42

N = number of reference points in each sub-set.
λ_0 and σ_λ = mean and standard deviation of the nodal longitude.
E_p = standard deviation expressed in pixels.
t_s and σ_t = mean and standard deviation of travel time and
E_l = standard deviation expressed in scan lines.

coordinates. As pointed out in section 2.3.1, half a scan-line error will affect t_σ by approximately 0,08 seconds. However the effect of pixel error on λ_0 varies with the distance from the centre of the image. Using reference points from the images of 27 and 28 July 1986, a parabolic equation was obtained:

$$\Delta\lambda_0 = 3{,}2508 \times 10^{-8}\bar{p}^2 - 1{,}4826\times10^{-5}\bar{p} + 1{,}0978 \quad , \qquad (2.26)$$

where $\Delta\lambda_0$ = the change in λ_0 resulting from a pixel error in the reference-point coordinates.

and $\bar{p}$ = distance from the centre of the image (in pixels).

b) If $\sigma_\lambda > \Delta\lambda_0/2$ and/or $\sigma_t > 0{,}08$ then the reference point farthest removed from the mean (defined by $\bar{\lambda}_0$ and $\bar{t}_s$) is deleted. This point is taken to be the reference point for which the function

$$f(\lambda_0, t_s) = \sqrt{[\{(\lambda_0 - \bar{\lambda}_0)/\Delta\lambda_0\}^2 + \{(t_s - \bar{t}_s)/0{,}1667\}^2]} \quad , \qquad (2.27)$$

is the largest.

c) $\bar{\lambda}_0$, $\bar{t}_s$, σ_λ and σ_t are computed for the remaining reference points and further points removed in the same way until the conditions for σ_λ and σ_t are satisfied.

When applied to the data in Table 2.6 the procedure removed 1,1,

3,0,4,2 and 4 reference points from the seven clusters respectively. This reduced the average line and pixel uncertainties (E_l and E_p) from 0,60 lines and 0,57 pixels to 0,42 lines and 0,33 pixels respectively.

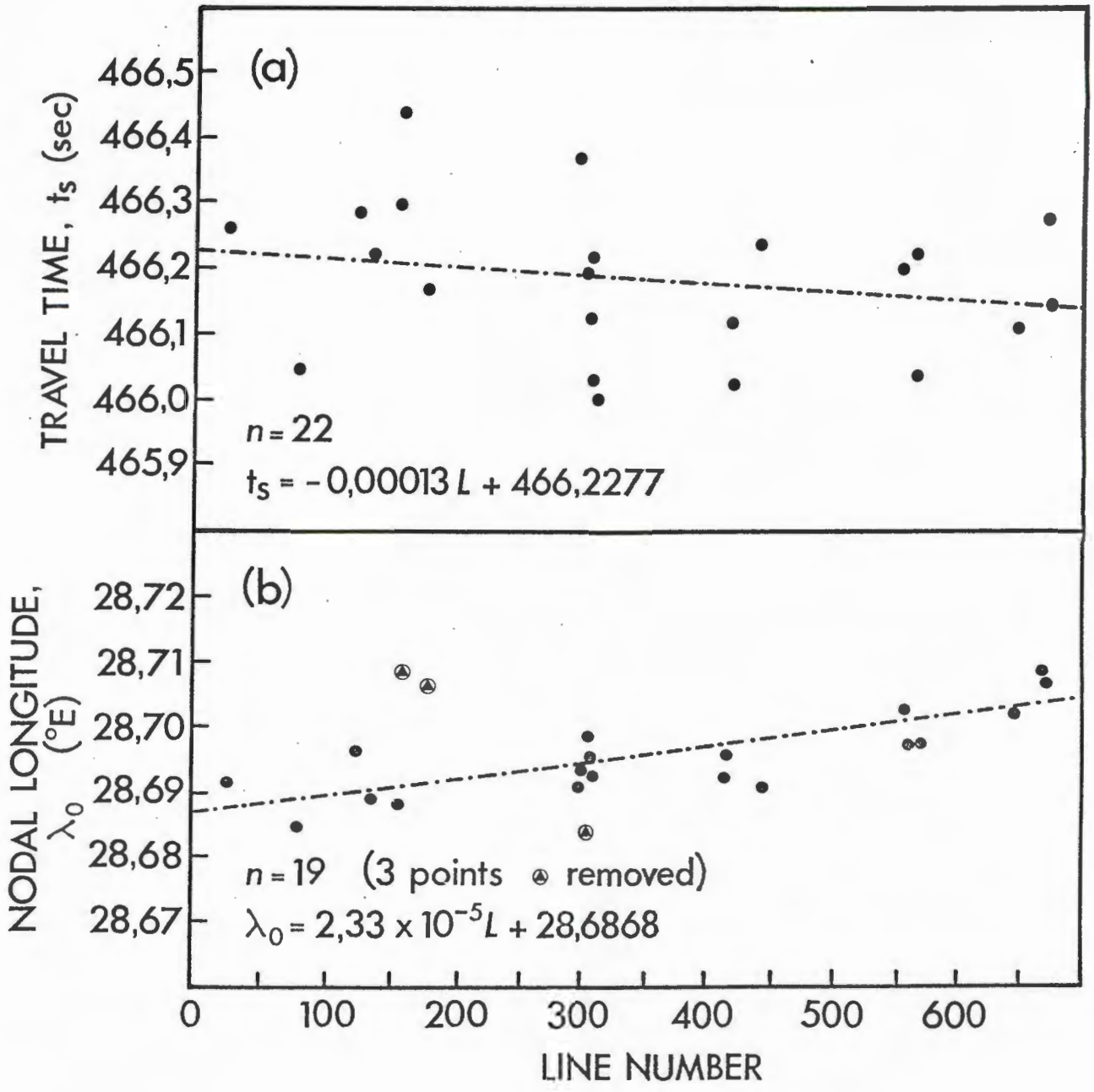


Fig. 2.12: Variation of t_s and λ_0 with scan line number.

Algorithm behaviour in the flight direction :

The variations of t_s and λ_0 as a function of scan line number was investigated through a subset of reference points chosen, as before, for a pixel spread of about 100 pixels (Fig. 2.12). The slope of the t_s linear regression is $-3,2x10^{-4}$ and that of λ_0, $9,1x10^{-6}$ - both being within the range found for the ascending pass (Table 2.4). These are equivalent to a one scan line error over 521 scans and one pixel over 1 099 scans. It can therefore be concluded that, as far as the along track behaviour of the model is concerned, similar responses are obtained for both ascending and descending tracks.

Algorithm behaviour in the scan direction :

Two subsets of reference points could be obtained from the image to study the algorithm behaviour in the scan direction. These were again selected to give a scan line spread of less than 100 lines. In the first set (Fig. 2.13) both t_s and λ_0 vary approximately linearly with pixel number. The slope of $2,53x10^{-5}$ for the λ_0 regression is similar to that of Figure 2.10 (ascending track), resulting in a 0,01° deviation (approximately 1 pixel) over 400 pixels. The two points beyond pixel number 1600, deviate from the linear regression suggesting a non-linear edge effect as was discussed in section 2.3.3. The t_s regression (Fig. 2.14a) show no appreciable nonlinearity and the slope is representative of a one-scan-line deviation over 630 pixels, which is a little more severe than was found in the case of the ascending track

image, but nevertheless of acceptable magnitude.

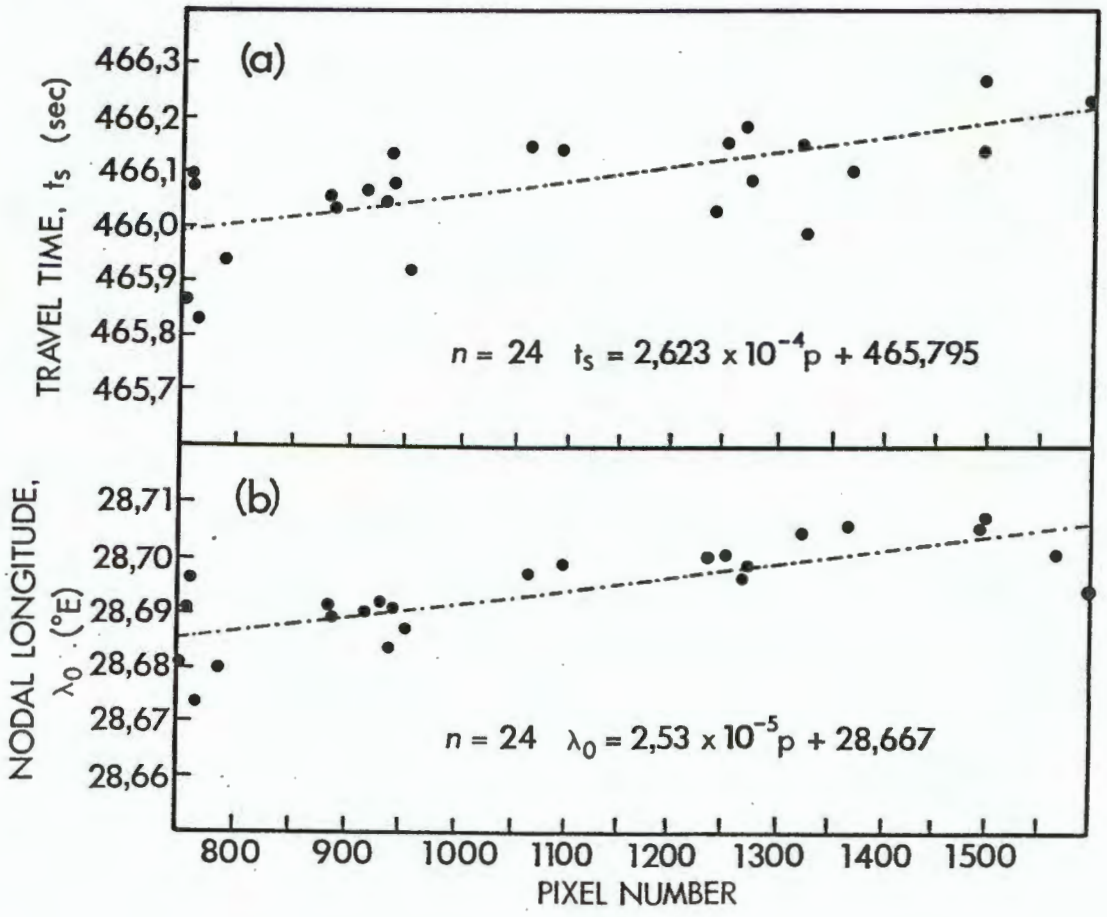


Fig. 2.13: Variation of t_s and λ_0 with pixel number. Sub-set-number one of reference points in the scan direction.

The second set of points in the scan direction (Fig. 2.14) span a wider pixel range than the first set. In this case the t_s and, in particular, the λ_0 regressions demonstrate appreciable non-linearity. The equivalent data set for the ascending track image also showed non-linearity in the λ_0 regression (Fig 2.11b), but the curvature was not as pronounced. This difference may be, as surmised in section 2.3.3, on account of the fact that the

change in the satellite attitude. Iterative adjustments of the scan angle indicate that a roll correction of 0,12° would remove the non-linearity.

The effect of the non-linearity on t_s is negligible, being equivalent to a deviation of one scan line over about 900 pixels, but upon λ_0 the effect is more serious. In the event of λ_0 being determined from a reference point near the centre of the scan and then used to compute the position for a point near pixel number 1900, the expected longitude error will be 0,035°. In the context of deriving sea surface motion, a feature being tracked will thus appear to be displaced westward and if two images, 24 hours apart are used an apparent velocity component of about 5 cm/sec will be introduced. If however only the centre portion of the image, say pixels 500 to 1500, is used , the deviation is reduced to 0,01°, or in terms of surface motion, to about 1,5 cm/sec. Such errors are acceptable.

2.4 TRANSFORMATION TO MERCATOR PROJECTION

To achieve the transformation to Mercator projection use was made of the image-to-image registration procedure available through the ARIES II image processing package. The essential step in this procedure is the computation of the transform functions which describe the relationship between the coordinates of the original image (the "slave") in terms of those of the corrected image (the "master"). The ARIES routine allows for the computation of

first, second or third order transform polynomials, where the third order functions are given by:

$$x' = a_0 + a_1x + a_2y + a_3x^2 + a_4xy + a_5y^2 + a_6x^3 + a_7x^2y + a_8xy^2 + a_9y^3 \quad (2.28a)$$

$$y' = b_0 + b_1x + b_2y + b_3x^2 + b_4xy + b_5y^2 + b_6x^3 + b_7x^2y + b_8xy^2 + b_9y^3 \quad (2.28b)$$

x', y' = pixel and line coordinates of the slave image

x , y = pixel and line coordinates of the master image.

If 10 or more ground reference points of coordinates (x_i', y_i') are identified in the slave image along with the corresponding master coordinates (x_i, y_i) then least-squares regressions of the form:

$$\hat{x}' = a_0 + a_1x + \ldots + a_9y^3 \quad . \quad (2.29a)$$

$$\hat{y}' = b_0 + b_1x + \ldots + b_9y^3 \quad . \quad (2.29b)$$

can be obtained. $\hat{x}', \hat{y}'$ are the slave coordinates from the regression. In the case of equation 2.29a the quantity $\Sigma(x' - \hat{x}')^2$ which is

$$\sum_{i=1}^{N} [x'_i - (a_0 + a_1x_i + a_2y_i + \ldots + a_9y_i^3)]^2 = E(a_j) \quad j=0,9$$

must be minimised. At the minimum $\partial E/\partial a_j = 0$ $j=0,9$

$$\therefore \partial E/\partial a_0 = \sum_{i=1}^{N} -2(x'_i - a_0 - a_1x_i - a_2y_i - \ldots - a_9y_i^3) = 0 \quad . \quad (2.30.1)$$

$$\partial E/\partial a_1 = \sum_{i=1}^{N} -2x_i(x'_i - a_0 - a_1x_i - a_2y_i - \ldots - a_9y_i^3) = 0 \quad . \qquad (2.30.2)$$

$$\partial E/\partial a_2 = \sum_{i=1}^{N} -2y_i(x'_i - a_0 - a_1x_i - a_2y_i - \ldots - a_9y_i^3) = 0 \quad . \qquad (2.30.3)$$

$$\vdots$$

$$\partial E/\partial a_9 = \sum_{i=1}^{N} -2y_i^3(x'_i - a_0 - a_1x_i - a_2y_i - \ldots - a_9y_i^3) = 0 \quad . \qquad (2.30.10)$$

These may be written as:

$$Na_0 + (\Sigma x_i)a_1 + (\Sigma y_i)a_2 + \ldots + (\Sigma y_i^3)a_9 = \Sigma x'_i \quad . \qquad (2.31.1)$$

$$(\Sigma x_i)a_0 + (\Sigma x_i^2)a_1 + (\Sigma x_i y_i)a_2 + \ldots + (\Sigma x_i y_i^3)a_9 = \Sigma x'_i x_i \quad . \qquad (2.31.2)$$

$$(\Sigma y_i)a_0 + (\Sigma x_i y_i)a_1 + (\Sigma y_i^2)a_2 + \ldots + (\Sigma y_i^4)a_9 = \Sigma x'_i y_i \quad . \qquad (2.31.3)$$

$$\vdots$$

$$(\Sigma y_i^3)a_0 + (\Sigma x_i y_i^3)a_1 + (\Sigma y_i^4)a_2 + \ldots + (\Sigma y_i^6)a_9 = \Sigma x'_i y_i^3 \quad . \qquad (2.31.10)$$

and solved for a_j, j=0,9 after having done the summations. The b_j, j=0,9 are obtained in a similar manner.

In view of the fact that all the mathematics above are already included in the ARIES routine, it was deemed desirable to use the routine even though its normal application requires the physical existence of a second image (which in this particular case does not exist) and the time-consuming, user-interactive identification of reference points on the two images. The problem of a second image could be overcome fairly easily by creating a blank image, overwritten with a latitude/longitude grid

conforming to the Mercator projection. Any number of ground reference points can then be obtained by using points on this grid and by computing the corresponding line and pixel numbers for the slave image, using the algorithms in section 2.2.5. However, a more efficient solution was offered by the ARIES routine's option for reading the ground control points from a file. Such a file could be prepared semi-automatically by a computer routine.

In a Mercator map projection, lines of longitude are equidistant and parallel. Lines of latitude are also parallel but the separation increases towards the poles. For any latitude, ϕ, the distance, d, from the equator is given by:

$$d = r_e.\log_e[\tan(\pi/4 + \phi/2).\{(1-e.\sin(\phi))/(1+e.\sin(\phi))\}^{e/2}]$$

(Snyder,1987) , (2.32)

Where r_e = earth's equatorial radius (6378,137 km)

e = eccentricity of the earth

$= [\sqrt{\{(r_e - r_p).(r_e + r_p)\}}]/r_e$

r_p = earth's polar radius (6356,752 km)

Distance, d, may more conveniently be expressed in terms of minutes of longitude by :

$$d' = (360x60xd)/(2.\pi.r_e) \quad , \qquad (2.33)$$

The program (GCPFIL, Appendix B) that was written to prepare the data file for use by the ARIES routine uses equation 2.33 and the

algorithms in sections 2.2.4 and 2.2.5 (Subroutines NOAN2 and NOAN3, Appendix A). The program essentially performs the following functions:

a) Use the algorithm in section 2.2.4 to compute the geographical coordinates for the corners of the slave image.

b) Inputs by the user of a scaling factor in the form of the number of pixels, in the master image, to be equivalent to one minute of longitude. This determines the dimensions of the master image.

c) Ground reference points are computed along lines of latitude.

The user is prompted for the latitudes to be used, as well as the number of control points to be computed along each line.

d) For each of the specified latitudes the program finds a longitude (to the nearest 0,1°) inside and nearest the western edge of the image. It finds a similar point near the eastern edge and then spread the remaining points evenly in between. For each of these positions the line and pixel numbers are computed - representing the slave x',y'-coordinates. The corresponding master image x,y-coordinates are computed from the scaling factor and equation 2.33. The coordinates are transferred to the data

file.Interpolation of points between the western- and eastern edges is done in terms of longitude, which has the desired effect of giving more weight to the edges where the greatest degree of nonlinearity occurs.

Having set up the data file, the ARIES routine can be used with minimum user interference to compute the transform equations. The actual transformation itself is carried out using another ARIES routine which calculates pixel values for the master image from the corresponding neighbourhood in the slave image, the coordinates being defined by the transform equations.

The transformation procedure described above is not the most elegant one possible. It requires four separate steps ie. :

i) The navigation routine (NOANAV) should be executed to compute λ_0 and t_s and to create a data file with the satellite orbital data for use in step (ii).

ii) The routine GCPFIL must be executed to prepare a ground control point file for computation of the transform equations in step (iii).

iii) Execute the ARIES routine to compute the coefficients for the transform equations. These are stored in a polynomial transform file for use in step (iv).

iv) Finally the second ARIES routine is used to perform the

actual resampling to the Mercator projection.

Ideally it should be one process but in practice such a process could not be executed on the Institute's image processing system with its limited memory capacity. Besides, such a process would have necessitated the rewriting of some of the software tools already available and, finally, it should be noted that in retaining the separate routines, transformation to any other map projection than Mercator, is easily accomplished simply by replacing the equation 2.32 in GCPFIL by a relationship appropriate to the desired projection.

2.4.1 Testing the transform procedure.

The transformation is carried out in order to move information from one projection (the image) to another (Mercator). This process involves three separate mathematical relationships:

i) The relationship between the geographical coordinates (ϕ,λ) and the coordinates of the original image (x',y') given by the navigation algorithms.

$$(x',y') = f_1(\phi,\lambda) \quad , \qquad (2.34)$$

ii) The cartographic relationship between geographical coordinates (ϕ,λ) and the coordinates of the final image (x,y)

$$(x,y) = f_2(\phi,\lambda) \quad , \tag{2.35}$$

This relationship is essentially given by equation 2.32 and is contained in program GCPFIL.

iii) The statistically derived relationship between (x,y) and (x',y') given by equation 2.28

$$(x',y') = f_3(x,y) \quad , \tag{2.36}$$

The behaviour of the navigation routine was discussed in section 2.3. The relationship (2.35) is based upon a model of the earth and is therefore not exact but errors are likely to be small and will be ignored here. This leaves the relationship (2.36), which is the set of polynomial transform equations, to be investigated.

For testing the polynomial transform functions, the ascending track image of 27 July 1986 was used. Sixteen ground reference points were identified in a small area centred upon line 1104, pixel 529 (overall line range = 1083-1123, and pixel range = 524-540). Two points were removed through the procedure outlined in section 4.3.4 and the remaining 14 used to compute the mean nodal longitude and travel time: $\lambda_0 = -14,5784°$ ($\sigma_\lambda = 0,0028$) and $t_s = -368,8233$ sec ($\sigma_t = 0,0714$)

Three questions needed addressing i.e.:

(i) Order of the polynomial to be used.

(ii) The number of control points required.

(iii) Variation of transform precision with position within the image.

2.4.1.1 Transform precision: The effects of order of the polynomial and number of ground control points

A 300 line x 300 pixel sub-image was created with its centre coinciding with that of the group of reference points used to compute λ_0 and t_s. The program GCPFIL was then used to compute 15, 20, 25, 30, 35 and 40 control points respectively from which, in each case, a 1st, 2nd and 3rd order polynomial was computed. A separate data set consisting of 25 control points, not coinciding with those used for construction of the polynomials, was also compiled. This data set consists of pairs of (x', y') and (x, y) coordinates, where the x, y-coordinates had been calculated using the cartographic relationship (exactness assumed) and x', y'-coordinates from the navigation routine (errors reduced to less than half a line and pixel by choice of image dimensions and proximity to reference location. Refer to tests in section 2.3) By applying any one of the polynomials to this data set, another estimate of the master image coordinates (x, y) is obtained and by comparing these coordinates with those in the test set, the error inherent to the transform equation is evaluated. The results obtained from the various combinations of transform order and number of control points used, are shown in Table 2.7

From the results in Table 2.7 it is clear that, with the

exception of the case where only 15 control points was used, the

Table 2.7: Standard errors (root-mean-square) for the master image line and pixel numbers calculated for 25 positions in a 300x300 test image. (5 control points per line of latitude)

Order of transform		Number of control points used. 15	20	25	30	35	40
1	Pixel error	3,94	3,97	3,96	4,02	3,99	3,96
	Line error	0,36	0,36	0,37	0,38	0,37	0,36
2	Pixel error	0,47	0,50	0,48	0,46	0,46	0,46
	Line error	0,24	0,22	0,24	0,23	0,22	0,23
3	Pixel error	7,48	0,27	0,26	0,23	0,22	0,19
	Line error	11,00	0,25	0,26	0,24	0,24	0,23

3rd order function produced the best approximation. It would also seem that error magnitude is fairly independent of the number of control points used - provided the number is larger than 15. The latter observation was further checked by repeating the 3rd order polynomial calculations for a 500 x 500 image centred on the same line/pixel as before (Table 2.8). While line errors remained essentially the same as in the smaller image, pixel errors increased. Error magnitude, as before, was only slightly reduced by increasing the number of control points. The pixel error obtained with 30 control points is noticeably smaller than the others but can not be regarded as indicative of an optimum number. The least squares procedure will tend to force the transform function through the control points hence errors will be a minimum in the vicinity of these points as is demonstrated by Table 2.8.Therefore since the positions of the 36 test points

vary relative to those used to compute the transforms, some random fluctuation in error size should be expected.

Table 2.8: Standard errors for the master image line and pixel numbers calculated for 36 positions in a 500x500 test image. Also shown in brackets, are the errors obtained from the control points used to compute the transforms themselves. Results are for a 3rd order polynomial.

	Number of control points.				
	20	25	30	35	40
Pixel error	0,42	0,44	0,29	0,40	0,37
	(0,28)	(0,35)	(0,27)	(0,30)	(0,25)
Line error	0,25	0,22	0,22	0,23	0,20
	(0,16)	(0,19)	(0,20)	(0,18)	(0,21)

Table 2.9: Standard errors in the master image lines and pixels computed for 500x500 subareas at various positions across the width of the NOAA AVHRR scene. All subareas centred on line 1104.

Central pixel	Pixel range	Pixel error	Line error
424	175 - 674	0,61	0,20
624	375 - 874	0,30	0,25
824	575 - 1074	0,18	0,24
1024	775 - 1274	0,18	0,22
1224	975 - 1474	0,19	0,26
1424	1175 - 1674	0,24	0,24
1624	1375 - 1874	0,62	0,30

2.4.1.2 Transform precision : The effect of position within the swath

To investigate the variation of transform precision with position within the image, 500 x 500 subareas were taken at regular intervals across the width of the scene; 3rd order transforms were computed using 35 ground control points (5 points on 7 lines) and the standard errors computed for 36 test points (points not used in the computation of the transform equations).

It is seen (Table 2.9) that the line error fluctuates around 0,25 lines, irrespective of position. The pixel error is small and fairly constant within the pixel range 375 - 1 674, but then increases rather rapidly towards the edges of the scene. Since one pixel in the master image equals one minute of longitude - by virtue of the scale chosen in GCPFIL - an error of 0,6 is approximately equal to 1 km which will introduce a spurious velocity component of about 1 cm/sec when using images 24 hours apart for derivation of advection vectors. Such an error would be small but not negligible, compared with expected ocean current velocities. An attempt was therefore made to try and reduce the pixel error, near the edge, to the same level as the line error. Two possibilities were considered: (a) increasing the number of control points used for computation of the transform and (b) reducing the width of the sub-image.

The effect of varying the number of control points was tested, using a 500x500 image centred on pixel 424 (the first position

in Table 2.9) and control points selected as (i) 6 points on 6 lines, (ii) 7 points on 5 lines, (iii) 8 points on 5 lines and (iv) 9 points on 5 lines. The test, however only confirmed the results previously obtained (Table 2.7) ie. that number of control points had little effect upon the pixel or line errors.

The effect of sub-area dimension on the error is shown by Table 2.10, which compares the error size for 3 different images all centred on the same pixels. In Figure 2.15 the pixel errors from the table are plotted as a function of sub-image position (central pixel number). The diagram indicates a precision gain with decreasing image width, but the gain is not really very dramatic. For instance, a standard error of 0,3 is predicted for a sub-image 500 pixels wide and situated between pixels 390 and 1 660. For a sub-image 300 pixels wide, the same error is

Table 2.10: The standard error in master pixel and line, for three different size sub-images at various positions within the scene. Errors computed for 3rd order transform functions derived from 35 ground control points.

	500x500		400x500		300x500	
Central pixel	Pixel error	Line error	Pixel error	Line error	Pixel error	Line error
224	-	-	-	-	0,62	0,25
324	-	-	-	-	0,39	0,23
424	0,61	0,20	0,41	0,24	0,25	0,20
624	0,30	0,25	0,18	0,25	0,22	0,24
824	0,18	0,24	0,20	0,20	0,19	0,22
1024	0,18	0,22	0,20	0,24	0,15	0,29

obtained when positioned between pixels 300 and 1 750. It would nevertheless be advantageous to use a smaller image size under circumstances where low velocity advection need to be studied near the edge of the image - say, pixel numbers smaller than 500 and larger than 1 500.

2.4.1.3 The overall transform error : slave to master image.

In section 2.4.1 it was pointed out that the transform procedure comprises three independent mathematical components i.e. (a) the navigation routine which describes the relationship between the latitude/longitude and the line/pixel numbers of the initial image (slave image), (b) the cartographic function which provides

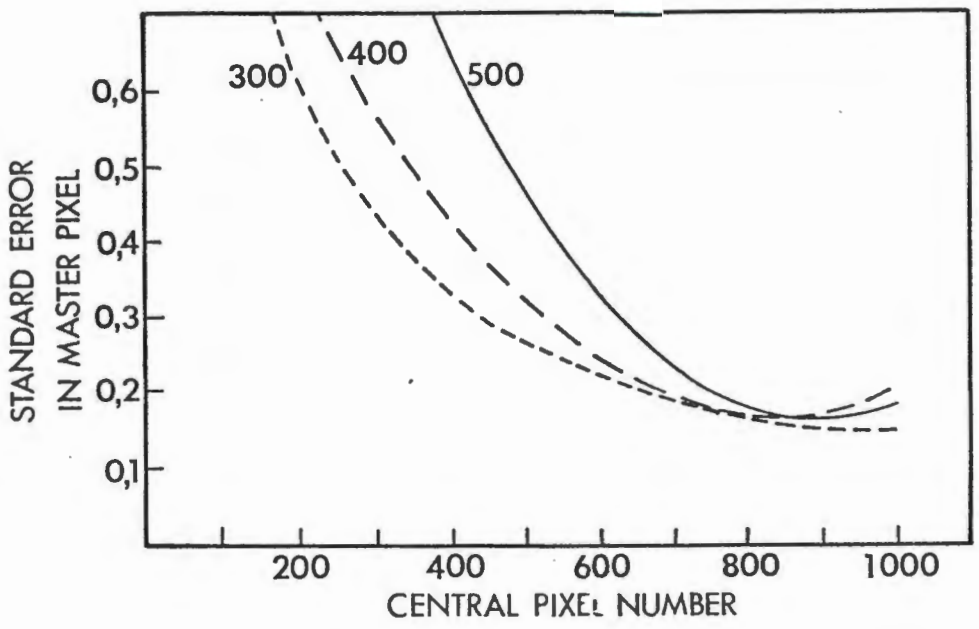


Fig. 2.15: The standard error in the master pixel for 3rd order transform functions. Three different size sub-images (500x500, 400x500, 300x500) at various positions within the AVHRR scene.

the relationship between latitude/longitude and the line/pixel-coordinates in the Mercator projection (master image) and (c) the statistically derived transform function which describes the relationship between the slave image line/pixel-coordinates and master image line/pixel coordinates. The behaviour and precision of the (a) and (c) components have been discussed in preceding sections, while a discussion of the cartographic relationship (b) is regarded as superfluous. However, it is still necessary to evaluate the performance of the procedure as a unit ie. to evaluate the cumulative error from all three components. In practical terms this means the transform procedure is expected to move an object in the slave image to a specific line/pixel position in the master image, as determined by its latitude and longitude. However, on account of imprecisions within the components of the transform, the object is likely to appear to have shifted from its expected position in the master image and eventually such an offset will appear as a spurious velocity component during the derivation of the sea surface advection velocities.

To facilitate this check, the images available in the Institute's archive were scrutinised for suitable scenes where (a) the satellite's track passed through Southern Africa and (b) was fairly cloud free in certain sections of the sub-continent. Five such scenes were eventually selected (Table 2.11). For each of these images the nodal longitude (λ_0) and standardised travel time (t_s) were computed using the group of reference points near Cape Columbine (32,83°S, 17,85°E). The same sub-image, as defined by

corner latitudes/longitudes and approximately 500 lines by 500 pixels was then taken from each image. Third order polynomial transform functions were computed using 36 ground control points. The selected sub-area contained 24 of the 145 dams used for testing the navigation algorithm (section 2.3) and, depending on cloud cover, up to 42 reference points could be identified from these and were used to check the accuracy of the transforms.

Two methods were used to express the transform errors:

(i) in terms of image coordinates, and

(ii) in terms of geographical coordinates.

Notation:

(x', y') : image coordinates (pixel/line) of reference points in the slave image.

(x,y) : image coordinates for the master image corresponding to (x',y')

(ϕ, λ) : geographical coordinates (lat./long.) in the master image corresponding to (x, y)

(ϕ', λ') : geographical coordinates for the reference points.

Method 1:

(a) Use the polynomial transform equations to compute (x, y) from (x', y')

(b) Use (ϕ', λ') in the Mercator relationship (eq. 2.32 and 2.33) to compute (x_0, y_0) for the master image. These are

regarded as the "true" pixel/line-coordinates for the particular reference point.

(c) Compute the RMS line error from $(y_0 - y)$ and the RMS pixel error from $(x_0 - x)$.

Table 2.11: Information on images used to check transform to Mercator projection. (All NOAA 9 day time ascending track images.)

Date	Reference points N	Mean pixel	Mean line	λ_0	t_s	Test area Pixel Range	Line Range
86-7-27	17	529	1102	-14,5800	-368,816	941-1500	691-1260
86-9-1	17	874	1134	-11,0754	-371,082	1251-1800	701-1380
86-11-23	16	1112	823	- 8,9300	-429,488	1461-1900	401-1100
87-4-30	17	1035	1075	- 9,6107	-384,126	1421-1860	641-1390
88-2-2	17	712	845	-12,5505	-416,081	1151-1700	401-1050

Method 2:

(a) Use the polynomial transform equations to compute (x, y) from (x', y')

(b) Use (x, y) and the inverse of the Mercator relationship to compute (ϕ, λ).

(c) Compute the RMS latitude error from $(\phi' - \phi)$ and the RMS longitude error from $(\lambda' - \lambda)$.

The results obtained for the five images are listed in Table 2.12 and indicate a maximum for the line error as 0,57, and the maximum pixel error as 0,92, the latitude error as 0,0082° and longitude as 0,0154°. In terms of distance over the surface the

latitude error is equivalent to 0,91 km and the longitude error (at 31°S) equivalent to 1,45 km.

The deviations for the individual test points demonstrated a fairly consistent positive bias in all four parameters (i.e. line, pixel, etc.), but negative deviations also occurred. Hence, for purposes of deriving advection velocities by comparing two separately transformed images, it must be assumed that a

Table 2.12: Root-mean-square (RMS) errors for the line, pixel, latitude and longitude for five test images.

Image Date	D	N	RMS Error Line	Pixel	Lat.	Long.
86-07-27	822,4	38	0,49	0,90	0,0071	0,0149
86-09-01	772,1	37	0,39	0,71	0,0057	0,0119
86-11-23	640,8	30	0,33	0,65	0,0048	0,0107
87-04-30	707,3	38	0,57	0,77	0,0082	0,0128
88-02-02	814,5	36	0,48	0,92	0,0069	0,0154

N = number of test points used to calculate RMS errors.
D = average distance from reference points to test points (in pixels).

positive error in one image may combine with a negative error in the other - giving rise to total errors which may be double in size to those shown by Table 2.12 or in absolute measures, 1,82 km in the north/south direction and 2,90 km in the east/west

direction. In terms of velocity components computed from pairs of images 24 hours apart, the velocity uncertainty represented by these numbers are respectively 2,1 cm/sec north/south component and 3,3 cm/sec east/west component.

These tests represent fairly extreme conditions, in that the areas transformed were large (generally more than 500 pixels by 500 lines) and, in addition, as far removed from the reference location (Cape Columbine) as was practically possible; Table 2.12 shows that the average distance D, varied between 640 and 822 pixels. From the preceding analysis of the behaviour of the navigation and transform procedures (Fig 2.11 and 2.14) it is known that the errors will increase with distance from the reference location and with size of the area transformed. Such a tendency is in fact detectable in Table 2.12. It therefore seems reasonable to accept these estimates as being representative of the upper limits of velocity errors. Indeed for most practical applications of feature tracking over the continental shelf, where the outer limit of interest would be within 500 km from the reference points on the coast, the errors may be reduced to less than half the figures considered here. On the other hand, it should be borne in mind that if the offset errors (km) are regarded as fixed, then the velocity error becomes inversely proportional to the time difference between the pair of images so that the 3,3 cm/sec east/west error component, for example, would increase to 6,7 cm/sec if image pairs 12 hours apart, in stead of 24 hours apart are to be used.

3. FEATURE TRACKING

Given a pair of sequential images, transformed to a common map projection, manual extraction of surface advection vectors becomes a fairly simple task. The image processing system's 'split screen' display facility can be employed to view both images simultaneously. The operator can then pick out small features such as eddies or other fine structures in the thermal features of the first image and attempt to match these with the features in the second image. If successful, the image coordinates (x_1,y_1) and (x_2,y_2) in the two images represent respectively the start and end points of the advection vector for that particular parcel of water. This procedure was used to estimate flow velocities in the Luderitz upwelling plume (Agenbag and Shannon 1988) and in a warm water ring from the Agulhas Current (Duncombe Rae *et al.* 1992a). It was found that identification of the features in the second image - ie. the end points of the vectors - is considerably eased if the vectors are drawn on the image as they are being defined. This eliminates the need to record the x,y-coordinates and speeds up the process; but the main advantage is that the operator can start by tracking the most prominent features first, thus obtaining a general picture of surface motion which serves as a guide in determining vectors from more obscure features.

Being able to observe the development of the vector field adds

interest to what would otherwise be an extremely tedious task and helps to speed up the extraction process. However, it remains a very time consuming process; for example, 260 vectors were derived to describe the flow associated with an Agulhas ring (Duncombe Rae *et al.* 1992a), in what amounted to a fairly small image section, and the process took most of two working days. The time factor therefore provides strong incentive for finding a means of automating the vector extraction process but there are other aspects as well. The most important of these being the question of accuracy. Thermal features in the ocean tend to be large and ill defined, which generally causes no real problem in identifying the pair of gross features, but identification of the exact pixels representing the vector end points is far from easy. This difficulty is alleviated by image processing techniques such as contrast enhancement, but an element of imprecision remains. A solution to this problem would be to extract vectors from several points on the same feature and use the average, but this would add substantially to the already very tedious nature of the task. The automated process based on the concept of 'template matching', to be discussed in the following section, however embraces this very principle and should in theory improve accuracy.

3.1 MATCH MEASUREMENT IN AUTOMATING FEATURE TRACKING.

The stratagem for automating the feature tracking procedure was

touched on in the introduction to image registration (Chapter 2). It was said that the idea is to match a small section of the first image in the sequence to a similar size section of the second image. This procedure, known as 'template matching' or 'prototype matching', is widely used in the field of digital image processing for purposes of pattern recognition and to automate the definition of ground control points used for geometric transformations of, eg. LANDSAT imagery. (In engineering terminology the process is referred to as 'match filtering'). When employed for feature tracking the template is represented by the small section from the first image in the sequence and is, in principle, tested against all possible subsections - of similar size- of the second image. In practice however, if the template is centred on coordinates (x,y), only the neighbourhood of this point in the second image needs to be tested since a water parcel could only have been transported a maximum distance of n pixels and/or m lines during the time interval between the two images. The section of image contained within the pixel range (x-n, x+n) and the line range (y-m, y+m) will thus be referred to as the 'search window' , where n and m will be determined by the template size, the interval between images and the maximum expected advection velocity for the oceanic region under study. The crucial element of the procedure is of course the determination of the match position and for this purpose a function is required which measures the degree of similarity between the template and the various section of the search window - called the 'match function'.

In the literature the concept of match measurement is often referred to as 'image correlation' on account of the fact that match measurement is most commonly performed through a particular mathematical expression known as the 'cross correlation function'.

3.1.1 The correlation function.

This important measure of match is derived from the 'distance measure' :

$$\sum_x \sum_y (f-g)^2$$

where f(x,y) is the template and g(x,y) the corresponding section in the search image and x and y the pixel and line numbers (Rosenfeld and Kak, 1982). This measure - which is actually a measure of mismatch - expands to:

$$\sum_x \sum_y (f-g)^2 = \sum_x \sum_y f^2 + \sum_x \sum_y g^2 - 2\sum_x \sum_y fg \qquad (3.1)$$

and it is clear that if $\Sigma\Sigma f^2$ and $\Sigma\Sigma g^2$ were fixed then the mismatch measure, $\Sigma\Sigma(f-g)^2$, would be large only when $\Sigma\Sigma fg$ is small and $\Sigma\Sigma fg$ - the cross correlation of f and g - may then be used as a match function.

In practice, the template is fixed, hence $\Sigma\Sigma f^2$ is fixed, but since the template is shifted over the search image, $\Sigma\Sigma g^2$ varies with position and the cross correlation as such cannot be used. Rosenfeld and Kak (*op. cit.*) makes use of the Cauchy-Schwarz inequality :

$$\sum_x \sum_y fg \leq \sqrt{\left[\sum_x \sum_y f^2 . \sum_x \sum_y g^2\right]} \qquad (3.2)$$

to show that the 'normalised cross correlation' :

$$R_{fg} = \frac{\Sigma\Sigma fg}{\sqrt{\Sigma\Sigma g^2}} \qquad (3.3)$$

may be used as a match function whose values range between o and $\sqrt{\Sigma\Sigma f^2}$.

The application of the normalised cross correlation function involves a very large number of computations. If template, f, is a square of dimensions m×m and the search window, g, a square of dimensions n×n then there are n^2 possible positions for f in g for which the function must be evaluated and for each position, m^2 pointwise multiplications of f and g elements must be performed. The same number of computations must be carried out for the denominator (eq. 3.3). It is possible to reduce the labour of computing the correlation by making use of the

convolution theorem which shows that if $F(\mu,\nu)$ and $G(\mu,\nu)$ are the Fourier transforms of the template, f(x,y), and the search window ,g(x,y), respectively, then the correlation of f with g is given by the inverse transform of $F(\mu,\nu)G^*(\mu,\nu)$ where $G^*(\mu,\nu)$ is the complex conjugate of $G(\mu,\nu)$ (Gonzales and Wintz, 1977). The pointwise multiplication of F and G^* requires f(x,y) to be expanded to the same dimensions as g(x,y) by the addition of extra lines and columns - eg. of value zero. The relative efficiency of the direct method or the frequency domain procedure using a fast Fourier transform (FFT), depends upon the number of non-zero elements in f(x,y) as well as the type of computer used. Estimates performed by Cambell (1969, quoted by Gonzales and Wintz, 1977) suggest that for 132 or less non-zero elements the direct method is the more efficient.

The normalising factor in equation 3.3 is not ordinarily computed when using the FFT procedure. Rosenfeld and Kak(1982) shows that ambiguous results may be obtained without the normalisation, but Gonzales and Wintz(1977) suggest that the normalisation only serves to sharpen the peaks in the array R_{fg} (eq.3.3) and the cross correlation has been used without normalisation as a match measure by eg. Leese *et al.*(1971) and Svedlow *et al.*(1978).

A variation of the normalised cross correlation function was suggested by Leese *et al.*(1971) and referred to as the 'cross

correlation coefficient' :

$$R_{fg} = \frac{Cov(f,g)}{\sqrt{\sigma_f^2}.\sqrt{\sigma_g^2}} \qquad (3.4)$$

where for a given test position of template ,f, against the corresponding section of the search window ,g, :

Cov(f,g) = the cross covariance of f and g

$$= \frac{\Sigma\Sigma\,(f-\overline{f})\,(g-\overline{g})}{N^2} \qquad (3.5)$$

$\overline{f}$ = template mean and $\overline{g}$ = mean for corresponding section of the search window.

σ_f^2 and σ_g^2 = variances of the template and corresponding search window section.

The cross correlation coefficient , which may be expanded to :

$$R_{fg} = \frac{[N^2 \underset{x\,y}{\Sigma\Sigma} fg - (\underset{x\,y}{\Sigma\Sigma} f)(\underset{x\,y}{\Sigma\Sigma} g)]}{\sqrt{[N^2 \underset{x\,y}{\Sigma\Sigma} f^2 - (\underset{x\,y}{\Sigma\Sigma} f)^2]} \cdot \sqrt{[N^2 \underset{x\,y}{\Sigma\Sigma} g^2 - (\underset{x\,y}{\Sigma\Sigma} g)^2]}} \qquad (3.6)$$

has been compared with other match functions (Svedlow *et al.* 1978) and demonstrated to be the most efficient for measuring translational motion if certain assumptions regarding noise are made. The function was originally employed by Leese *et al.*(1971) to track cloud motion and has since been applied to the derivation of ice motion (Ninnis *et al.* 1986) and sea surface advection (Emery *et al.* 1986, Garcia and Robinson 1989, Tokmanian *et al.* 1990) - it is also the match function adopted, in this study, to test the automatic extraction of sea surface advection vectors and in the following section the practical application of this will be explored in greater detail.

3.1.2 Application of the cross correlation coefficient.

If ,in the application of an automated tracking procedure, the template, f(x,y), is taken to be a N-pixel by N-line array and the search area, g(x,y), an array of dimensions MxM (where N<M) then it is easily seen that the number of relative shift positions , ie. the number of times the match function needs to be evaluated , is given by :

$$Sn = (M-N+1)^2 \qquad (3.7)$$

In the evaluation of equation 3.6 the two terms, $\Sigma\Sigma f$ and

$\sqrt{[N^2\Sigma\Sigma f^2-(\Sigma\Sigma f)^2]}$, are constants and need to be computed only

once for the given template. All other terms, however, depend on g(x,y) and therefore must be computed for all relative shift positions. A quick calculation shows that evaluation of equation 3.5 for the NxN and MxM arrays would involve approximately $(M-N+1)^2.(2N^2+7)$ multiplications and $(M-N+1)^2.(4N^2+2)$ summations. If applied to a full AVHRR scene this would imply billions of operations and even when the number of relative shifts are reduced by defining g(x,y) a small subset of the total image, it still remains a formidable computation task.

As was pointed out in the preceding section, (3.1.1), the cross correlation of f(x,y) and g(x,y) may be computed through use of a FFT procedure - possibly with considerable savings in time. The Cov(f,g) term in equation 3.4 may also be evaluated by this means; however, neither the normalised cross correlation (eq. 3.3) nor the cross correlation coefficient (eq. 3.4) can be computed in the frequency domain and has to be computed after performing an inverse Fourier transform (Schowengerdt 1983).

In their work on cloud motion, Leese *et al.*(1971) actually assumed σ_f and σ_g (eq. 3.4) to be constants for a given

search window and indicated that it was not necessary to go beyond computation of the covariance matrix. Ninnis *et al.* (1986) applied the cross correlation coefficient to tracking of pack ice motion in the Beaufort Sea, using NOAA AVHRR imagery divided into 22x22-pixel templates and 32x32-pixel search windows. The window means were subtracted and R_{fg} computed from the inverse transform of $F^*(\mu,\nu)G(\mu,\nu)$. The variances σ_f and σ_g were computed in the spatial domain with σ_g being an array of values corresponding to all possible positions of the template within the search window. An identical matching procedure was employed by Emery *et al.*(1986) to derive sea surface advection velocities off the British Columbian coast. They found the method could not be applied directly to the AVHRR sea surface temperature (SST) images, but was successful when the images were first converted to images of SST gradients. This result was in agreement with Svedlow *et al.*(1978) who demonstrated that a gradient operator applied to LANDSAT imagery improved the success rate of automated matching procedures. The gradient operator used by Emery *et al.* (1986) was suggested by Van Woert(1982) and referred to as the 'temperature gradient magnitude (TGM)' and defined as :

$$|\nabla T(x,y)| = \frac{1}{2\Delta h}\sqrt{[\{T(x-\Delta h,y)-T(x+\Delta h,y)\}^2+\{T(x,y-\Delta h)-T(x,y+\Delta h)\}^2]} \qquad (3.8)$$

Both Van Woert (*op.cit.*) and Emery *et al.*(1986) first smoothed the temperature field with a two dimensional moving average filter before computation of the TGM.

3.2 IMPLEMENTATION OF AUTOMATIC FEATURE TRACKING.

In this section the practical implementation of an automatic feature tracking procedure, using the cross correlation coefficient as a measure of match, is described. Implementation involves creation of a computer program which will execute as efficiently as possible within the constraints of the available computer system. Unfortunately, in this case, the constraints imposed by the Institute's small image processing system are so severe that much of the discussion in this section cannot be regarded as of general interest. For example, one of the principal reasons for automation is to speed up the process of extracting the advection vectors. Hence computational efficiency is of great importance and immediately brings to the forefront the question of whether to adopt the simpler, but less efficient, digital implementation of equation 3.6 or the more complex, but more efficient, FFT implementation. The latter seems to be the obvious route to go, but it demands storage space for several two-dimensional complex arrays which, in our case, is very difficult to meet. For purposes of this study, both methods were tried (sections 3.2.1 and 3.2.2) and in both cases the structure of the program had to be tailored to the capacity of the machine. This was particularly so in the case of the FFT procedure and it

was in the end found that in the process of coping with the large storage demand, the advantage of speed was lost and the direct digital procedure actually turned out to be the faster. This would probably not be the case with larger systems where memory capacity is not such a restriction.

3.2.1 Automatic feature tracking using two-dimensional fast Fourier transforms.

This procedure, embodied in the program 'ADVECT' (Appendix C) and using repeated applications of a one-dimensional fast Fourier transform (FFT) algorithm (Sartori-Angus 1984), was tested first as an implementation of the procedure described by Emery *et al.*(1986).

Commensurate with the FFT requirement for the array dimensions being 2^n and the need to deal with large velocity vectors associated with the Agulhas Current, it was deemed essential that the routine should be able to handle up to 64x64 search arrays. This meant 32768 bytes of memory was needed to store one of the three complex arrays used to perform the FFT and the image processing system did not have the addressable memory capacity to hold more than one such an array. Temporary storage of intermediate results was therefore done in the system's 'image memory array' which, though faster than disk storage/retrieval, was nevertheless a lot slower than direct memory operations.

Although a detailed description of program 'ADVECT' is not considered necessary, some light may be cast upon practical problems related to implementation of the automated procedure by briefly discussing the main components of the program ie. the four subroutines ADVEC1, ADVEC2, ADVEC3 AND ADVEC4.

Subroutine ADVEC1 : This routine

(a) obtains the filenames of the two images to be used in the feature tracking process. These are assumed to be NOAA AVHRR , calibrated thermal band images (normally band 4), converted to TGM using equation 3.8 and transformed to Mercator projection. Land and cloud should be masked with value 255.

(b) obtains the desired template and search window dimensions (maximum = 64).

(c) obtains a window shift distance. This is the number of pixels/lines by which the top-left corner of the search window is shifted for successive vector extraction attempts - ie. it determines the potential spatial density of the vector field.

(d) creates a file for storing the coordinates of the vector start and end points.

Subroutine ADVEC2 : This routine

(a) creates three dummy 8-bit images to serve as data storage areas for the results of the Fourier transforms.

(b) creates a 16-bit image and loads the template and search images - each of which occupies 8 bits.

Subroutine ADVEC3 : This routine

(a) reads template and search window data from the image #4 (the 16-bit image). In the process it checks for the presence of land and cloud pixels (value 255); these are set to zero in the output array and if the number of such pixels, in either window, exceeds 1% of the total, that window position is abandoned. The template is symmetrically padded with zero's to the same dimension as the search array.

(b) computes the averages for each of the windows and the variance for the template window.

(c) computes an array of variances for the search window - corresponding to all possible positions of the template within the search window. These represent the g-variance term in the denominator of equation 3.6 . Since the template has even-valued dimensions a central element does not exist in the array and the variance, for any given location of the template, is written in the bottom-right pixel of the four surrounding the geometrical centre. Alignment with the covariance matrix is achieved by transposition of the output ADVEC4. The array of variances is stored in image number 3 (one of the three 8-bit images).

(d) the template and search arrays are stored as complex numbers in images 1 and 2 (the other two 8-bit images created by ADVEC2)

after subtracting the respective window averages and conversion to $^{\circ}C.Km^{-1}$ - conversion is however normally not required if the TGMs are computed with the same Δh (eq.3.8).

Subroutine ADVEC4 : This routine

(a) computes the Fourier transforms, $F(\mu,\nu)$ and $G(\mu,\nu)$, for the template and search window arrays stored in images 1 and 2. This is done by using subroutine 'FFT1' for repeated one-dimensional transforms of the rows and columns of the two-dimensional arrays. No shift of origin is performed.

(b) carries out a pointwise multiplication of $F^*(\mu,\nu)$ - the complex conjugate of the template transform - with $G(\mu,\nu)$ to obtain $R(\mu,\nu)$.

(c) computes the inverse transform of $F^*(\mu,\nu)G(\mu,\nu)$ and divide by the f-variance and g-variance terms (denominator of eq.3.6). The inverse transform is done as one-dimensional transforms, using FFT1, on the rows and columns of $R(\mu,\nu)$. In order to achieve alignment of the resulting covariance matrix with the g=variance matrix, the first and second halves of both the rows

and columns of the covariance matrix is swapped around ie. for a search window of dimensions NxN :

$$r(i_1, j_1) \rightarrow r(i_2, j_2)$$

where $i_2 = i_1 + N/2$. If $i_2 > N$ then $i_2 = i_2 - N$

and $j_2 = j_1 + N/2$. If $j_2 > N$ then $j_2 = j_2 - N$

This procedure correctly located the end points of translation vectors in a pair of test images consisting of a template image of temperature gradient magnitude (TGM) and a search image consisting of the same data but subjected to a linear shift. Further development was however curtailed when it became apparent that it would take about 7 minutes and 15 seconds to process a single 64x64 search area with a 20x20 template.

The 'successive-doubling' algorithm used for the FFT is not as efficient as algorithms using tables to store the transform coefficients, suggesting at least one way of reducing processing time, but with array storage needs already stretching the computing system's capabilities, it seemed unwise to experiment with procedures which would create additional demands. Instead, it was decided to test the performance of a procedure based upon a direct digital implementation of the cross correlation coefficient.

3.2.2 Automatic feature tracking based upon digital computation of the cross correlation coefficient.

The denominator of the equation for the cross correlation coefficient (eq. 3.6) may be rewritten as :

$$N^4\sqrt{\left[\frac{N^2\Sigma\Sigma f^2-(\Sigma\Sigma f)^2}{N^4}\cdot\frac{N^2\Sigma\Sigma g^2-(\Sigma\Sigma g)^2}{N^4}\right]} = N^4\sqrt{(C_f.C_g)} \qquad (3.9)$$

Where C_f and C_g are now recognisable as the variances of f and g (the template and search window arrays) respectively. And since the variance of a series remains unchanged when the average of the series is subtracted, subtraction of the window means, as suggested by Ninnis *et al.*(1986), will not affect the denominator of equation 3.6 . Also, assuming f(x,y) and g(x,y) are images of temperature gradient magnitude derived by application of equation 3.8 (using a fixed Δh) to the two images after transformation to a common Mercator scale projection, then the conversion of the pixel counts of f(x,y) and g(x,y) to absolute measures, say $^{o}C.Km^{-1}$ simply involves multiplication by a constant K. In which case the denominator becomes :

$$N^4\sqrt{(K^2C_f\ .\ K^2C_g)} \quad or \quad K^2N^4\sqrt{(C_f.C_g)} \qquad (3.10)$$

In the numerator of equation 3.6, multiplication by conversion factor K, and subtraction of the window means will have the effect that the term, $\Sigma\Sigma f.\Sigma\Sigma g$ become equal to zero. If in the remaining term, $N^2\Sigma\Sigma fg$, f is replaced by $Kf-K\bar{f}$ and g by $Kg-K\bar{g}$ ($K\bar{f}$ and $K\bar{g}$ being the window means) then :

$$
\begin{aligned}
N^2\Sigma\Sigma fg = {} & K^2(N^2\Sigma\Sigma fg-\bar{g}\Sigma\Sigma f-\bar{f}\Sigma\Sigma g+N^2\bar{f}\bar{g}) \\
& -K^2\left(N^2\Sigma\Sigma fg-N^2\Sigma\Sigma f.\frac{\Sigma\Sigma g}{N^2}-N^2\frac{\Sigma\Sigma f}{N^2}.\Sigma\Sigma g+N^4\frac{\Sigma\Sigma f}{N^2}.\frac{\Sigma\Sigma g}{N^2}\right) \\
& -K^2(N^2\Sigma\Sigma fg-2\Sigma\Sigma f.\Sigma\Sigma g+\Sigma\Sigma f.\Sigma\Sigma g) \\
& -K^2(N^2\Sigma\Sigma fg-\Sigma\Sigma f\Sigma\Sigma g)
\end{aligned}
\qquad (3.11)
$$

From equations 3.10 and 3.11 it is seen that working in absolute units and/or subtraction of the window means would not affect the cross correlation coefficients. Consequently these operations were dropped when programming to create 'AUTOTR' for feature tracking using the digital application of equation 3.6 .

Program AUTOTR (Appendix D) is similar in design to ADVECT, consisting of 4 subroutines , AUTOT1, AUTOT2, AUTOT3 and CRCOEF, which perform virtually the same functions as their equivalents in ADVECT:

Subroutine AUTOT1 : Only minor modifications to ADVEC1 were

needed to create this subroutine. The template and search window dimensions are limited to 32 and 64 respectively - effectively 31 and 63 since the dimensions are required to be odd.

Subroutine AUTOT2 : As in ADVECT, the image pair is loaded into video memory as a 16-bit image to speed up data extraction. Since array storage is now less of a problem, no images are created to serve as data buffers.

Subroutine AUTOT3 : This routine reads the window data and checks for land/cloud pixels - the 1% limit for such pixels is retained. The window data are passed as integer arrays to CRCOEF. Real arrays would have been more efficient but could not be stored in memory.

Subroutine CRCOEF : The template is positioned in the top-left corner of the search window and then shifted to cover all possible positions and for each position equation 3.6 is evaluated. In this equation $\Sigma\Sigma f$ and $\Sigma\Sigma f^2$ are constants and need only be computed once. The terms, $\Sigma\Sigma g$ and $\Sigma\Sigma g^2$ lend themselves to a computation scheme of column summations where only the sum for the leading edge column need to be computed for each fresh position while shifting from left to right. The $\Sigma\Sigma fg$ term must, however, be computed in its entirety for each new position of the template. The best registration position is

given by the maximum of $|R_{fg}|$ where $|R_{fg}| \leq 1.0$ (Svedlow *et al.* 1978) and the advection vector taken from the centre of the search window to the centre of the template position for which $|R_{fg}|$ - maximum.

Program AUTOTR was tested against a pair of TGM images where the second image was a copy of the first but with the data subjected to a linear displacement. All vector end points were computed correctly with R_{fg} - 1.0 and processing time was reduced to 3 min. 50 sec. for a 63x63 search window and 21x21 template. Compared with the first program, this represented a huge improvement (47%) in processing speed but in absolute terms, it is still a very slow process - in fact it is comparable or even slower than the rate attainable through manual feature tracking. However, if the procedure could be shown to be capable of reliable derivation of advection vectors, it would still be a highly attractive method from the point of view of objectivity and being able to function without operator interaction.

3.2.3 Testing the automatic feature tracking procedure.

A pair of test images were prepared, using sections of the NOAA 9 ascending pass image of 27 July 1986 and the subsequent

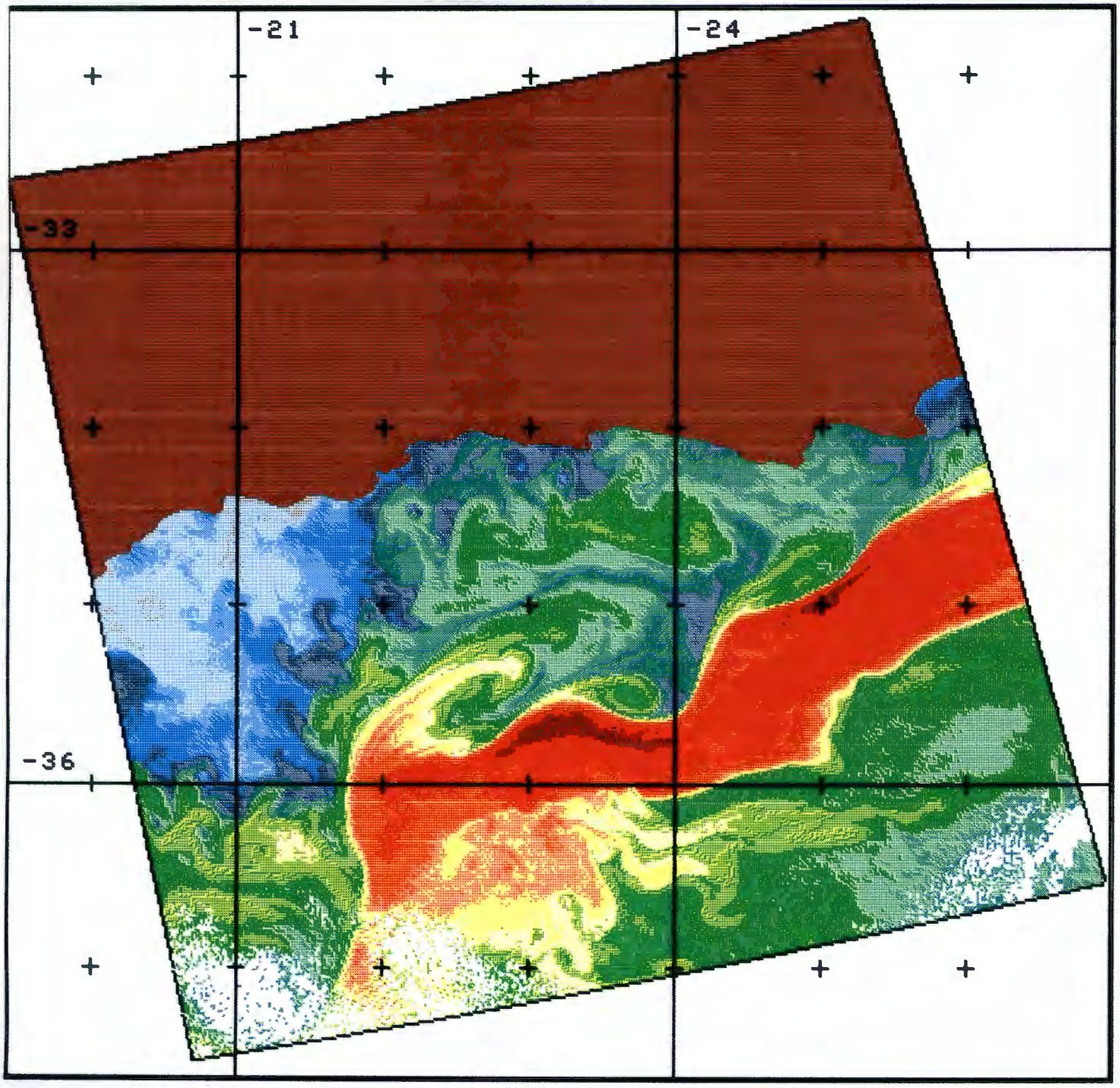


Fig. 3.1: The relative sea surface temperature structure in the image area used for testing the automatic feature tracking procedure. Image data from NOAA 9 AVHRR band 4, 27 July 1986. Colours follow the normal spectral sequence of blue, green, yellow, red with light-blue indicating the coldest and dark-red the warmest water.

descending pass, of the same satellite, approximately 11 hours and 23 minutes later on 28 July. The image section covered the South African south coast, from Cape Agulhas to Port Elizabeth and southwards across the Agulhas Current (Fig.3.1). The AVHRR band 4 images showed a great deal of thermal structure, produced by small eddies along the South Coast and larger shear-edge structures associated with the Agulhas Current. Band 4 data were calibrated and filtered with a 3×3 low-pass, mean-value filter as recommended by Emery *et al.*(1986) to remove low level variations. Thermal gradient magnitudes were computed with $\Delta h = 1$ (eq.3.8). Transformations to Mercator projection was carried out to a scale of 1' of longitude being equivalent to one pixel. Land and cloud were masked with a pixel value of 255. Using this TGM image pair, a set of 88 vectors were derived manually to serve as a reference for judging the performance of the automatic procedure (Fig.3.2). The origins of these vectors were stored in a file and subroutine AUTOT3 temporarily modified to compute vectors only for windows centred on these positions so that a 1:1 comparison with the manually derived set could be carried out. Since several of the vector base points were close to the land margin or near cloud, the window rejection limit of 1% was relaxed for the search window to 50% in an attempt to obtain as many comparisons as possible but also to investigate the effect of these conditions upon the method's performance.

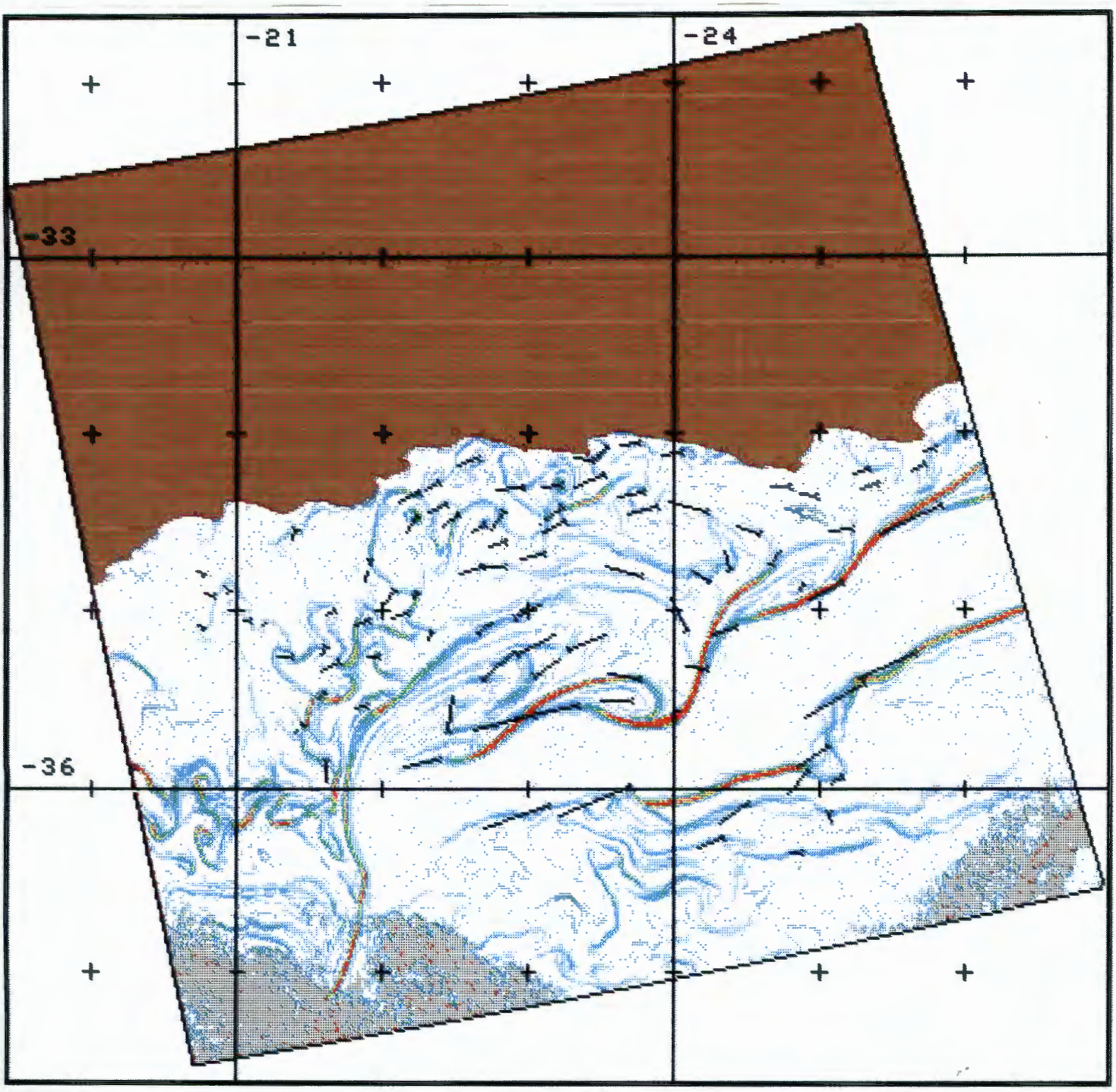


Fig. 3.2: The thermal gradient magnitude (TGM) pattern derived from Figure 3.1. The colour sequence is the same as in Figure 3.1 with high TGMs shown in red. Advection vectors derived from manual feature tracking are indicated as arrow symbols (the arrow has a feather on the tail). The tail end rests on the feature tracked and the size (length) is a direct reflection of the distance moved during the *ca.* 12 hours separating the image pair.

Inspection of the image suggested that a 17x17 template would, in most cases, contain sufficient structural detail to expect successful matching to be achieved. In places a smaller window, perhaps as little as 11x11, seemed sufficient. In their work, Emery *et al.*(1986) used a considerably larger template, ie. 22x22. It was therefore decided to run the test with a small range of window sizes , 13x13, 17x17 and 21x21, to try and determine the optimum. Manually extracted vectors ranged in magnitude from 1,4 (5 cm/s) to 31 (115 cm/s) with the majority between 5 (19 cm/s) and 15 (56 cm/s). The smallest vector is close to the uncertainty limit for the geometric correction procedure (section 2.4.1.3), but this is of no importance as far as this test is concerned. The very large vectors however, do create a problem in the sense that with the search window limited to 63x63, a vector with magnitude 31, cannot be derived unless it happens to be aligned with the diagonals of the window. In fact, with a 21x21 template, the largest vector which will still allow the template to be wholly contained within the search window, irrespective of direction, is one of magnitude 22. Six vectors larger than this were therefore removed from the test set, leaving 82.

The vectors from AUTOTR were compared visually with the manually derived ones and classified as 'poor', 'fair', 'good' or 'uncertain', depending on the degree of similarity. Although a great deal of care was taken in the manual tracking procedure, situations did occur where the AUTOTR vectors differed significantly from the manual set but neither could, with

absolute certainty, be identified as being the correct one. The results from this exercise (Table 3.1) were fairly similar for the three template sizes and not very good at all.

Table 3.1 : The degree of similarity between advection vectors derived through automatic feature tracking with program AUTOTR (using 3 different template sizes) and manually derived vectors.

TEMPLATE SIZE						
	13x13		17x17		21x21	
	Number	%	Number	%	Number	%
Poor	29	39,7	26	36,1	21	30,9
Fair	2	2,7	3	4,2	5	7,3
Good	31	42,5	30	41,7	31	45,6
Uncertain	11	15,1	13	18,0	11	16,2
Total	73		72		68	

The automatic procedure yielded fewer vectors due to rejection by the land/cloud-test in the program and also due to discarding those where dubious results could have been caused by the proximity of land or cloud. The probability for this to happen increased somewhat with increasing template size. The largest template also yielded the best results, mainly through reduction of the number in the 'poor' category. However, even when given all the benefit of doubt, the automated procedure failed completely in one out of three attempts.

In their work on ice motion, Ninnis *et al.* (1986), as well as the

sea surface advection study by Emery *et al.*(1986), vectors based upon a maximum cross correlation coefficient magnitude of less than 0,4 , were rejected. No such test was applied in this experiment with AUTOTR and could possibly account for the poor performance of the procedure. A frequency table of the correlation coefficients for the 'poor' and 'good' matches, however dispenses with this idea (Table 3.2).

According to this frequency table negative coefficients were invariable associated with poor matches even though Svedlow *et al.* (1978) pointed out that negative coefficients should still yield valid results. Removal of such vectors, however, does not

Table 3.2 : A frequency table relating the performance of the automatic feature tracking procedure (in terms of 'poor' and 'good' matches with the manual procedure) to correlation coefficient.

Correlation Coefficient	Template size					
	13 x 13		17 x 17		21 x 21	
	Poor	Good	Poor	Good	Poor	Good
R≤0,0	6		4		1	
0,0<R≤0,2						
0,2<R≤0,3			1		1	1
0,3<R≤0,4	1	1	3	3	4	7
0,4<R≤0,5	6	3	6	5	5	6
0,5<R≤0,6	10	9	5	9	3	7
0,6<R≤0,7	4	7	3	6	2	5
0,7<R≤0,8	1	5	2	6	3	2
0,8<R≤0,9	1	5	2	2	1	3
0,9<R≤1,0		1				

materially alter the situation. As for the remainder of the range of coefficients, only a slight shift in weight towards 'good' matches occurs with progressively increasing correlation coefficient and no cut-off point can be identified where the removal of incorrect vectors would not be offset by a loss of a similar number of good ones.

In order to establish circumstances giving rise to false vector results, window contents associated with those which received a 'poor' rating (13×13 template) were viewed and grouped according to, what seems to be, the four most common factors :

(a) Divergence/deformation/rotation (No. of false vectors = 8)
As can be seen in Figures 3.1 and 3.2 , divergent or rotational motion was the norm rather than the exception in the image pair used for the test. Rapid motion of this type, associated with the shear edge eddies along the fast moving Agulhas Current produced deformation of structures which precluded correct matching between the two images and was a major source of error near the Current boundary.

(b) Very strong TGM signals (No. of false vectors = 8)
This factor is related to (a) in the sense that it tended to occur primarily along the edge of the Agulhas Current with its strong thermal gradient. Whenever a section of this structure was present within the template, the correlation coefficient was completely dominated by the high pixel counts associated with the structure. The procedure then effectively tracked along the edge

of the Current, producing very high correlation coefficients when the high counts in the template overlaid high counts in the search window and the final maximum cross correlation value - which defines the end point - virtually became a matter of chance.

(c) Weak structures (No. of false vectors = 6)

The impression was gained that template windows without coherent TGM structures equivalent to gradients equal or greater than 0,5 °C/Km tended to yield false vectors.

(d) Chance (No. of false vectors = 7)

When seen within the confines of the template window, fragments of TGM structures often demonstrated very little detail and could bear a strong resemblance to adjacent, but unrelated, structures. Under such circumstances, chance matching with the wrong structure seems unavoidable. In a few cases the degree of similarity between unrelated structures (which actually possessed a considerable amount of detail) was quite surprising and only slight changes in the 'target' structure caused better matches to be achieved with the unrelated feature.

The automatic vector extraction procedure is based upon pattern recognition which, in turn, is based upon an assumption of shape invariance. In reviewing the four factors, (a) to (d), above, it seems that with the possible exception of factor (b), all errors stem in some manner or other from a violation of this principle. Wahl and Simpson (1990) modelled the effects of diffusion and

air-sea heat exchange on sea surface temperature (SST) patterns and showed that over a 12-hour period, spatially variant heat exchange, such as would be caused by the passage of cloud, would have a sufficiently large effect upon the SST to influence the cross correlation coefficient. Cloud was present along the western and southwestern edges of the two images used in the test, but the impression was gained that most of the test area was unaffected by cloud passage. This does not entirely rule out such a possibility though and some errors, especially those in classes (c) and (d) may conceivably have arisen from spatially variant heat exchange processes with the atmosphere. But, at best, this was probably a fairly minor effect compared with the shape distortion that was evidently brought about by shear-, rotational- and divergent flow patterns associated with the Agulhas Current.

The pattern recognition procedure, as applied here, was designed for simple advective motion and is incapable of dealing effectively with the more convoluted advection pattern. More sophisticated procedures which can, for example, handle rotational flow have been developed but at the cost of greatly increased computation time (Wahl and Simpson, *op. cit.*) - an unattractive prospect, considering the already extensive processing time required by the existing method. It would be very difficult, if not impossible, to implement such a procedure on the Institute's existing image processing system. A more practical approach, from this point of view, would be to limit the degree of distortion by reducing the time interval between

the pair of images used. In using NOAA AVHRR images, this is feasible since with two satellites in orbit, images may be obtained at intervals of 3-4 hours. However, in reducing the time between images, errors in the image navigation and geometric transformation procedures are emphasised and confidence in the resulting vector results undermined. In Chapter 2 (Section 2.4.1.3) it was concluded that velocity estimates may be biased by 3 to 7 cm/sec for a pair of images 12 hours apart. If the time separation is reduced to 4 hours, the bias will be 9 to 21 cm/s and could seriously corrupt the representation of most sea surface advection fields, including the one under consideration here where most vectors were between 19 and 56 cm/sec.

Conclusions :

At the beginning of this chapter a brief description was given of the manual tracking procedure. In the process, reference was made to the problems of vector accuracy and operator interaction as potential weaknesses of the manual method which were expected to be open to improvement by an automated procedure. By and large these expectations were not borne out by the experimental results, while at the same time certain disadvantages in the automatic procedure , not fully realised before, came to light through experience gained with it. There are five main points :

(i) Vector accuracy. The automatic procedure fared poorly and produced erroneous results in at least 30% of the cases. Admittedly, the data set used in the experiment presented the procedure with extremely difficult conditions, created ,firstly,

by the fast Agulhas Current and , secondly, by the fairly long time separation of 12 hours between the two images. Away from the edge of the Current the procedure was more effective and sometimes astonishingly so, giving the impression that in a different oceanic region, acceptable results may be expected - as was indeed reported by Emery *et al.*(1986). The Agulhas Current , however, is a physical structure of great importance to oceanographic studies around South Africa and, before an automatic procedure could be applied successfully, a means will have to be found to deal with the rotational and deformational effects associated with the Current. Some potential solutions have been considered within the discussion above, but were discarded for practical reasons.

(ii) Spatial resolution. The 'template' used in the pattern recognition procedure should be seen as a slab of water assumed to be transported without deformation. When this assumption is not satisfied, the process fails. Divergent motion on scales smaller than the template can therefore not be resolved. This may or may not be a disadvantage, depending on the application. In descriptive oceanography 'mean motion' or 'slab-like motion' is often a more useful quantity to measure than the small scale motion which may be regarded as noise, but the converse is true when physical processes are studied.

(iii) Operator interaction. The template matching procedure turned out to be computationally more expensive than anticipated. At a rate of 3 min 50 sec per window location (assuming a 63×63

search window), processing of a 512x512 image would take in excess of 16 hours if vectors are computed at intervals of 32 lines and pixels, giving a maximum of 256 vectors. This is slow but nevertheless automatic and frees the operator from the tedious manual extraction task. Unfortunately this gain is at least partially negated by: (a) The additional image preprocessing demanded by the automated procedure. For example, land and cloud masking, not needed for the manual procedure, is an absolute necessity in the case of the automatic procedure. Producing such masks for night-time images, for which no visible band data are available, can be particularly troublesome in regions of low sea surface temperature. (b) The need to edit the vector field produced by the automatic procedure.

(iv) The data boundary problem. With a 63x63 search window the automatic procedure cannot be used reliably within about 30 pixels (approximately 30 Km) from the coast. This is a major disadvantage since the nearshore region is important from the point of view of physical oceanographic processes such as upwelling and also from the point of view of it being the habitat of many important fish species and other marine biota. The same restriction is imposed by cloud masses and the edges of the image area. In the experiment , comparing the results of the automatic and manual procedures, the number of vectors obtained by the automatic procedure was 11 - 17% smaller than the number from the manual procedure.

(v) Time separation between images. Manual extraction of

advection vectors can often be accomplished, without much difficulty, using pairs of images 24 hours apart or possibly even more. In order to reduce the problems of rotational motion and structural deformation in the automated procedure, the maximum time separation seems to be about 12 hours. This reduces the available data base of images amenable to processing with the automated procedure. In regions such as the South African south- and southwest coasts the frequent passage of cloud formations associated with travelling atmospheric low pressure systems, create a problem with the acquisition of a suitably time-spaced pair of images. The reality of this problem was experienced when an attempt was made to obtain suitably cloud free images coinciding with acoustic doppler current profiler (ADCP) measurements made during November 1989 - only one pair of images separated by 24 hours could be obtained.

Of these five points, numbers (ii),(iv) and (v) are inherent to the principle of pattern recognition as embodied in the template matching procedure and impose restrictions which would generally apply to all attempts to construct sea surface velocity patterns through this type of automatic procedure. The other two are technique dependant and actually represent an open-ended problem where progressively better results can be expected with increasingly sophisticated procedures, until a stage is reached where reliability and savings in operator time are such that the advantages of the automatic procedure outweigh the disadvantages. Under circumstances such as ours, where chances for successful

implementation of a more sophisticated procedure is limited by the capabilities of the available equipment, the disadvantages posed by points (ii),(iv) and (v), argue against attempts to try and improve speed and reliability and consequently force a decision to adopt a procedure which leans more towards a manual rather than an automated technique.

3.3 IMPLEMENTATION OF A SEMI-AUTOMATIC PROCEDURE.

As an introduction to this chapter manual feature tracking was briefly described and the two main problem areas mentioned were ie. labour intensiveness and the difficulties with accurate identification of vector end points. At that stage it was anticipated that these problems could be solved by automating the process, but test results examined in the preceding section, indicated that neither problem was satisfactorily addressed by the procedure. However, in the process of experimenting with the automatic procedure it was realised that the principle of template matching could be incorporated into a 'semi-automated' manual tracking procedure where it would serve to eliminate the subjectivity involved in manual vector end point determination. When testing program AUTOTR (Section 3.2.3) the operator identified suitable features in the template image and provided the vector start point coordinates to the program which then used template matching to determine the end point. This process often failed due to the large area which needed to be searched and the consequent opportunity for a number of spatially related factors

to interfere with the pattern recognition process. However, when used in conjunction with manual tracking, the operator could indicate the approximate end point, thus drastically reduce the area to be searched and hence the chances of a false end point. In the introduction to this chapter it was also pointed out that visual identification of corresponding gross features in image pairs, seldom presented a problem, but that accurate identification of actual corresponding pixels was difficult. In other words, identification of the approximate end point is easily done, where-after the 'best fit' position may be obtained from the maximum cross correlation coefficient.

Incorporation of template matching into the framework of a manual tracking procedure is therefore expected to serve the dual purpose of increasing vector accuracy and reducing the labour involved. However, the greater part of the labour in manual tracking stems from more mundane tasks such as switching backwards and forwards between the template and search image displays, driving the display cursor to the desired positions, recording coordinates and drawing the arrow symbol representing the velocity vector. Automating these functions is technically not difficult but implementation in the form of a comprehensive computer program capable of performing all the functions related to manual feature tracking, presented a fairly formidable programming task, which finally crystallised in the form of program 'MTRACK'.

The vector computation procedure proposed here, thus, involves

only one mathematical concept viz. template matching using the cross correlation coefficient as the match function. Since this is also the principle on which the automated procedures are based and has therefore already been described in the preceding part of this chapter, a discussion of the theoretical aspects of the proposed method is not required. The practical implementation of the concept involves a computer program, the structure of which is largely determined by the available hardware, but as an example a listing of program MTRACK has been included in Appendix E and some detail about the main features of the program is provided below. The important part of the program - the part which deals with aspects of feature tracking - is found in the subroutine called MTRAC2 (the remainder of the program is of peripheral nature and probably not of much general interest).

The entire program architecture was designed to comply with the basic objective, namely to minimise the level of operator interaction, while providing maximum flexibility in terms of selecting mode of operation (ie. pointwise tracking or template matching), template size and search window size. The need for flexibility in the choice of operating parameters was implied by the test results obtained with the automatic procedure. These results showed clearly that the template matching method tends to fail under a variety of oceanographic conditions but, as was discussed above, one can expect the incidence of failures to be reduced by adapting template and/or search window dimensions to suit the dictates of the ambient conditions. In this manner the range of conditions over which this mode may be effectively

employed is expanded - a desirable achievement from the point of view of increased accuracy and reduced labour. However, it is to be anticipated that circumstances will occur where, due to the inherent characteristics of the principle of pattern recognition (section 3.2.3), this mode of feature tracking must be abandoned in favour of the pointwise procedure.

To evaluate the efficiency of MTRACK relative to the original manual tracking procedure, 25 vectors were selected from the set of 88 previously used to test program AUTOTR (Section 3.2.3). These vectors, representing a cross section of 'easy' and 'difficult' ones, were derived twice, first with MTRACK and then by the usual manual procedure. The times taken to derive the vectors were :

i) MTRACK (window mode) 27 min 10 sec

Manual tracking 34 min 10 sec

Time saving = 21 %

ii) MTRACK (cursor mode) 14 min 58 sec

Manual tracking 23 min 26 sec

Time saving = 36 %

After initial difficulty to identify the designated features, the operator learned to identify them with greater ease during the subsequent test runs, hence the sharp drop in processing time. It is nevertheless clear that, in spite of the learning effect biasing results in favour of the manual procedure, a time saving of somewhere between 20 and 40 % may be expected by using the

semi-automated procedure.

In the next chapter the semi-automatic procedure will be employed to compute advection vectors in three case studies - each of which exhibits a large range of sea surface advection velocities - and the results will be used to evaluate the general performance of the procedure and to assess the accuracy and precision of the computed vectors.

A description of program MTRACK.

Program MTRACK was written for the Institute's ARIES II image processing system and in order to give a rational description of the program some ARIES II terminology needs to be clarified :

Display topology : The video display area may be divided in several ways to display one or more images simultaneously. The 'split screen' mode displays two images side by side.

Image : An image is created in video memory (VMA) by specifying the data to be displayed in terms of number of lines and pixels. A pixel is displayed as either 8 or 16 bits. This is called 'ixel depth'. The first bit (bit 0) is used to display the cursor and for writing annotation on the display and is referred to as the cursor or annotation bit. The remaining 7 or 15 bits are used to display features or themes. Several images may reside in VMA simultaneously.

Feature : Normally refers to radiometric data eg. the five AVHRR channels would be five 'features'. They are stored on disk as 'feature files' where all features belonging to a particular scene has the same dimensions and a common 3-character 'area code'.

Theme : Typically refers to a classification such as a land- or cloud mask. Themes are stored in 'theme files' with the same area code as the features belonging to that particular scene.

Program MTRACK expects the user to have configured the video display for the spit screen mode and to have created at least two images (the template and search images). Each image should contain one or more features and at least one of them should also contain one or more themes.

The program is structurally complex, consisting of a main program and 22 subroutines to facilitate overlay loading of the program - an essential requirement in view of the limited memory capacity of the ARIES image processing system. However, only the main structure and functions will be discussed. More details may be obtained from the listing in Appendix E - the code is well annotated.

The main program, MTRACK, calls 4 lower level routines :

(i) MTRAC0 - to present the main user menu.

(ii) **MTRAC1** - allows the user to :

a) Select the images and features to be used for feature tracking.

b) Apply a contrast stretch to these images to assist in the identification of thermal features suitable for tracking.

c) To select the image and theme field to be used for displaying the arrows representing advection vectors.

d) To define the manner in which the arrows are to be produced. The arrow symbol is drawn as a straight shaft with a short cross bar representing a feather on the tail end. The arrow may be produced with the feather end on the start point (the point in the template) or with the shaft centred over this point. The size (length) of the arrow is defined by a user supplied scaling factor. A factor of 2,0 will, for example result in a length of twice the distance between the indicated start and end points.

e) A direct-access storage file is opened to store the coordinates of the extracted vectors.

(iii) **MTRAC2** - the program section which deals with feature tracking. It presents a menu of options but in essence performs two functions ie. vector definition and deletion of previously produced vectors. Vector definition may be done in two modes :

a) Cursor mode. In this mode the operator indicates the actual start and end points with the display cursor. The program switch the cursor to the template image and request it to be driven to the start point. Once selected this pixel is marked

with a dot and the cursor switched to the search image and the request is repeated for the vector end point. The dot in the template image helps with the identification of the corresponding pixel in the search image. The arrow symbol is then drawn in the annotation field and the operator must indicate acceptance or not. If accepted, the arrow is redrawn 'permanently' in the theme field and the coordinates saved in the direct access storage file. If not accepted, the symbol is removed and the process restarted.

b) Window mode. The procedure in this case is similar to 'cursor mode' except that the user only needs to indicate the approximate end point in the search array, upon which template matching is used to determine the best match point. Windows are set to default dimensions of 13x13 and 23x23 but may be changed interactively to maxima of 17x17 and 27x27 respectively. Frames corresponding to the selected window dimensions are drawn around the indicated points in the two displays.

In an attempt to compensate in part for structural deformation of features, the operator may rotate the search image window relative to the template orientation. If this option is selected the window symbols are replaced by cross-hairs which may be interactively rotated until a satisfactory alignment with the image features is achieved. Once a rotation angle is selected it remains in effect until it is changed again, and the search image window is drawn with this rotation. The

template window data (since it is the smaller set) is resampled commensurate with the specified rotation before template matching is executed.

Vectors are removed by driving the cursor to the start point of the arrow symbol. The arrow is deleted and the data record, in the direct access storage file, marked as deleted.

(iv) MTRAC4 - saves the defined vectors in a disk theme file and computes the vector position in geographical coordinates. It then uses these coordinates to compute the absolute speed and direction.

Default selections are presented in most places where the operator is required to make a keyboard entry - these are selected by pressing the 'ENTER' key. This means the operator generally only needs to drive the cursor and press ENTER during the feature tracking process. It is also possible to step backwards through the program hierarchy by pressing 'CONTROL-Z'.

4. APPLICATIONS

In this final stage of development of a procedure for the derivation of sea surface advection velocities from series of thermal images, three case studies will be descibed in which the semi-automatic feature tracking procedure is applied. Firstly, it is a practical test whereby subjecting the procedure to a range of oceanographic conditions, limitations and flaws may be exposed. Secondly, it can be seen as a pure scientific exploration of advective processes in the ocean. With these ideas in mind, the case studies were chosen on the basis of the following criteria :

a. Each of the case studies should represent a situation where a large range of velocities could be expected.

b. In each case several fairly cloud free images, covering a range of time intervals, should be available.

c. Each case study should portray an event in the South East Atlantic, adjacent to the South African and Namibian west coasts, and thus be considered as being of scientific interest, particularly regarding potential consequences for the commercial fisheries along these coasts.

In addition to these case studies, an attempt was made to acquire a set of images coinciding with one of the Sea Fisheries Research Institute's (SFRI) cruises during which the Acoustic Doppler Current Profiler (ADCP) was employed so that the two sets of current measurements could be compared. The ADCP was installed in one of the SFRI's vessels (R.S.Africana) towards the end of 1989 and only a small number of comprehensive surveys have as yet been carried out (Boyd *et al.*1992). As was mentioned in the previous chapter, only one pair of moderately cloud free images could be found, coinciding with a section of one of the ADCP surveys along the South African south coast. These images were 24 hours apart and although an attempt was made to carry out feature tracking, the exercise was defeated by the combination of patchy cloud and the large time interval being inherently incompatible with the highly dynamic Agulhas Current regime.

The subsequent case study analyses have shown however that even had it been possible to derive an adequate set of vectors, quantitative comparison with the ADCP data would have been difficult because of the convoluted nature of the velocity fields observed over much of the case study scenes. As a consequence when two observations are even a short distance apart, it becomes difficult to tell whether differences are due to errors in the procedure or merely reflect the spatial variations of the velocity field. This problem was also encountered when an attempt was made to evaluate the precision of the vector computation procedure based on the reproducibility of two independently derived vector sets. Such evaluations were carried out in case

studies one and two and the results will be described as part of the respective discussions.

Each of the case studies also provided one or more opportunities where the derived data could be compared with advection velocities previously reported in the literature. In order to accomplish this without continually interrupting the flow of the text describing case studies, the available knowledge on surface advection patterns will be described in the following 'Oceanographic background' section. Direct comparisons will nevertheless be pointed out in the case study discussions and summarised in the final 'summary and conclusions' section.

4.1 THE OCEANOGRAPHIC BACKGROUND.

The three case studies to be presented cover, in combination, a section of the South East Atlantic Ocean stretching northwards from the Subtropical Convergence (*ca.* 39°S) to 24°S (north of the Lüderitz Upwelling Centre) and westward from the Agulhas Bank (*ca.* 21°E) to about 5°E. Surface advection in this region is affected by three major current systems : The Agulhas western boundary current, the Antarctic Circumpolar Current and the Benguela eastern boundary current (Fig. 4.1).

4.1.1 The Benguela Current System.

The Benguela Current is an equatorward flow of relatively cold

surface water, up to 200 km in width and situated between the South Atlantic subtropical gyre and the African west coast between about 15° and 35°S (Shannon 1985). (Note: since this investigation is only concerned with the southern half of the Benguela, subsequent references to the Benguela Current will imply the area as far north as 25°S). Water from the Agulhas Bank and Agulhas Current enter this region from the south so that the southern boundary may perhaps be more appropriately defined as the Agulhas retroflection (Shannon *et al.* 1981), rather than the southern extremity of the continent, as implied by the given definition. In the west it is hard to differentiate between the northward flow of the Benguela and the similarly directed South Atlantic subtropical gyre. The presence of cool upwelled water in the Current generally contrasts with the adjacent warm oceanic water, but since upwelling is variable in both space and time, the Current boundary is not satisfactorily defined through temperature either (Bang 1971).

While wind induced upwelling does not serve to define the Current's boundaries, it is never the less the most important characteristic of the Benguela. The driving force for upwelling along the western coast of southern Africa is mainly provided by the South Atlantic high pressure system which is

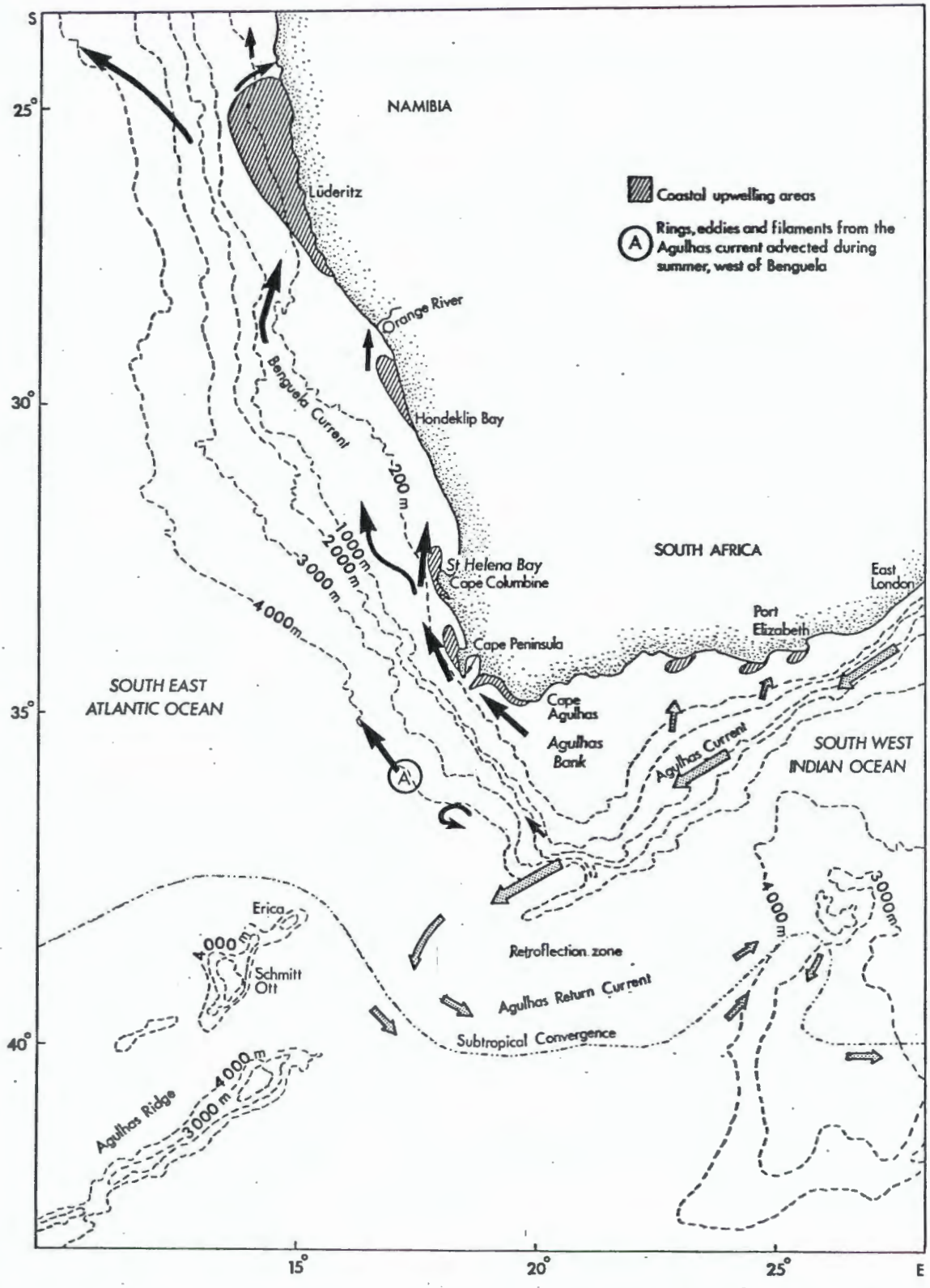


Fig. 4.1: The study area. A conceptional representation of important physical features and ocean currents constructed after figures presented by Shannon(1985) and Walker (1986).

maintained throughout the year. The anticyclonic winds associated with the high pressure, are guided by the continental escarpment and thermal barrier produced by the desert-like nature of the coastal plain (Nelson and Hutchings 1983), to create upwelling favourable conditions throughout the year. During winter the high pressure shifts northwards, giving rise to a spring/summer upwelling maximum and autumn/winter minimum (Shannon 1985). The seasonal fluctuation is most pronounced in the south where the macro-scale wind field is modulated by the eastward passage of cyclones spawned in the belt of westerly winds between 35 and 40°S (Nelson and Hutchings 1983). These travelling cyclones have relatively little effect on winds north of about 31°S and upwelling is perennial in this region.

Five upwelling cells have been recognised in this part of the Benguela upwelling system, ie. the Cape Agulhas, Cape Peninsula, Cape Columbine, Namaqua (or Hondeklip Bay) and Luderitz cells, the locations of which generally correspond with narrow parts of the continental shelf (Fig. 4.1; Lutjeharms and Meeuwis 1987). Although water upwelled in these cells tend to merge in to a continuous band of cold water along the coast, the individual cells are usually easily identified on most thermal- and visible band satellite imagery of the area, in the form of filaments or plumes stretching in
a general westerly direction from the upwelling front. Lutjeharms and Stockton (1987) defined filaments
as narrow protuberances extending more than 50 km from the main thermal upwelling front and being narrower than 50 km. Plumes are

larger structures, which are formed either through coalescence of filaments, as is often found in the Hondeklip and Luderitz cells; or as single entities off capes, such as Cape Peninsula or Cape Columbine. The Lüderitz cell not only produces the coldest water but also has the largest extent. The average offshore extent is 380 km (Lutjeharms and Meeuwis 1987) but a filament extending more than 1300 km from the coast has been reported (Lutjeharms, Shillington and Duncombe Rae 1991). Although the incidence of upwelling has been shown to decrease sharply from north to south (Lutjeharms and Meeuwis 1987), inspection of thermal imagery shows the existence of upwelling plumes and/or filaments at all sites throughout the year. This does not however imply active upwelling at all times; Lutjeharms and Meeuwis (*op. cit.*), for example, found no active upwelling in the Cape Peninsula cell during winter.

The Cape Peninsula upwelling cell, while possibly one of the least active of the five locations in the southern Benguela region, has associated with it an important oceanographic feature: a well-developed front and equatorward jet (Duncan 1967; Bang and Andrews 1974). The front, which forms the western boundary of the Cape upwelling plume, is situated over the shelf break, approximately 70 km west of the Cape Peninsula. The surface flow recorded by Bang and Andrews (1974) was 60 - 80 cm/s, increasing to more than 120 cm/s in the 100 to 180 meter depth range. These authors suggested that the jet served to transport warm Agulhas Current water northwards into the Benguela system and that way intensify the front and help to maintain it

for long periods - up to a month after upwelling favourable winds have switched off.

There is in fact good reason to believe that the Cape upwelling front is maintained - at least at subsurface levels - throughout periods of downwelling and even throughout winter (Brundrit 1981; Hutchings, Holden and Mitchell-Innes 1984). The jet associated with the front may therefore be regarded as the fastest permanent current feature known to exist in the Benguela system. Fast equatorward flow also occurs off Cape Columbine but the structure is more complex and some uncertainty still exists as to the kinematics of this area. Shannon (1985) reviewed available data and suggested that two fronts and two jets exist to the north of 33°S ie. (i) the main oceanic front and jet with an estimated surface velocity of 60 cm/s near the 400 m isobath and (ii) another front immediately west of the Columbine upwelling plume (200 m isobath) with which is associated a shallow time-variable jet. Both equatorward and poleward-directed velocities of up to 60 cm/s have been recorded in the nearshore jet. Shannon postulated the existence of a divergence zone near 17° 30'E between the two branches of the current but was unable to establish with certainty whether both branches are present at once. The existence of a nearshore poleward surface flow however seems to be well established now. Duncan and Nell (1969) deduced from driftcard studies that such a current existed between from al least 32°S to Table Bay (just north of the Cape Peninsula) during all seasons and extended southward around the Cape Peninsula during winter. Using drogue tracking and current

profiling, Nelson (1985) also found nearshore poleward flow to the north of the Peninsula and extending southward around the Peninsula during the upwelling relaxation phase. Nearshore poleward flow around the Columbine peninsula was documented by Lamberth and Nelson (1987) from drogue tracking data and by Holden (1987) from moored current meter data. Many of the above features were also noted in data from the ADCP studies (Boyd *et al.* 1992).

The description of flow features given above does not constitute a complete discussion of this aspect of the Benguela system, but does represent the more important elements from the point of view of relevance to the mixed species purse-seine fishery on the South African west coast. This multi-million Rand fishery has been, since 1966, primarily dependent on catches of anchovy (*Engraulis capensis*) which spawns mainly on the Agulhas Bank during the spring and summer months and recruit along the West Coast during the subsequent autumn and winter (eg. Crawford, Shannon and Pollock 1987). With the bulk of the anchovy catches consisting of such recruits, coupled with indications that the species is being exploited close to or in excess of the maximum sustainable yield, the fishery is extremely sensitive to environmental factors. The fish may select the Agulhas Bank for spawning on account of the warmer water generally present there, an important factor for normal egg development (King *et al.* 1978). Feeding conditions on the Bank are unfavourable for young larvae, as shown by persistently low Chlorophyll levels (Shannon *et al.* 1984). Larval survival is therefore expected to be

strongly dependent on successful migration to the more productive west coast regime. Nelson and Hutchings (1987) estimated the cruising speed of larvae (1 to 2 weeks old) at 1 - 3 cm/s, juveniles (10 weeks) at 2 - 4 cm/s and adult anchovy at 20 - 30 cm/s. It is therefore clear, from the discussion above, that the longshore currents would normally assist in the westward and northward transportation of larvae and juvenile fish while the adults, on their return journey to the spawning ground, would benefit from the poleward flow inshore. However, since the northward migration would be facilitated by the shelf-edge jet, on the seaward side of the Cape Peninsula and Cape Agulhas upwelling plumes, recruitment may be adversely affected by frontal excursions, such as filaments, or eddy formation which could result in an offshore transport and possible starvation. Brundrit (1981) has, for example, shown that the shelf-edge jet, off the Cape Peninsula, is close to instability and that small perturbations could rapidly develop into mesoscale eddies. Distribution patterns of anchovy eggs in fact strongly support the concept of recruitment losses through such mechanisms (Hampton 1987). Other pelagic species such as pilchard (*Sardinops ocellata*), and the less economically important round herring (*Etrumeus whiteheadi*) and horse mackerel (*Trachurus capensis*) may be similarly affected.

4.1.2 The Agulhas Current.

The Agulhas Current is on average 90 km wide (Pearce 1977) and generally warmer than 22°C. It flows with surface velocities exceeding 2 m/sec (Gründlingh and Lutjeharms 1979) southwards and westwards along the east coast of southern Africa, roughly following the 200m isobath, until it encounters the Agulhas Bank to the south of Port Elizabeth. At this point it separates from the coast to follow an undulating course southwestward along the 200m isobath, off the eastern edge of the Bank. The Current separates from the Bank near 36-37°S but generally continues on a southwesterly course until it retroflects at 19°E (±1,5°) (Harris, Legeckis and Van Foreest 1978) to flow eastward as the Agulhas Return Current, directly north of the Subtropical Convergence. The Agulhas Current, as such, is mostly located outside the indicated geographical area relevant to this investigation and will therefore not be discussed further. It is known however that Agulhas Current water intrudes into the South East Atlantic and Benguela system, sometimes along the shelf edge as filaments and also in the form of anticyclonic rings. The latter spawned in the retroflection zone and advected northwestward into the Atlantic. These features are not only of general interest to the oceanography of the region, but of particular relevance to all three case studies to be presented.

The intrusion of warm water, classified as 'Agulhas water' by thermohaline characteristics, into the regions north and west of the Cape Peninsula has been known since the work of Darbyshire

(1963, 1966) and Shannon (1966). The influx seems to be most prominent during summer and is thus thought to be related to the prevailing winds - ie. advanced by anticyclonic circulation and *vice versa*. Shannon (*op. cit.*) described the intruding water as 'shallowing tongues of 21 - 22°C penetrating the Atlantic Ocean'. These tongues are usually found more than 180 km offshore. Modelling of the Current as wind driven circulation also related leakage of Agulhas water to the Atlantic to characteristics of the wind field over the Indian Ocean. The models could also show a northwestward branching of the leakage (De Ruijter 1982, Boudra and de Ruijter 1986). Warm filaments of variable dimensions are commonly seen on satellite thermal imagery. They generally seem to originate near the southern tip of the Agulhas Bank and pass in a northwesterly direction around the Cape Peninsula into the southern Benguela. The trajectories of such filaments were traced by Lutjeharms and Stockton (1987) who confirmed a preference for filaments to move along the western shelf edge of the Agulhas Bank towards the Cape Peninsula upwelling cell. Such filaments are generally about 50 to 100 m deep (Chapman *et al.* 1986, Lutjeharms and Gordon 1987). The origins of the filaments, as mapped by Lutjeharms and Stockton (1987) varied widely : from the southeastern edge of the Agulhas Bank to the most westerly position of the Agulhas retroflection near 16°E. As was mentioned above, the Agulhas Current separates from the coast near Port Elizabeth; large meanders often occur down stream of the separation point, with cyclonic plumes of warm water generally associated with the crest of the shoreward meander (Lutjeharms, Catzel and Valentine 1989). Plumes emanating from meanders just

south of the southern tip of the Agulhas Bank, have been observed to drift northwestwards (Lutjeharms, Catzel and Valentine *op.cit.*) and thus present at least one mechanism for the formation of Agulhas filaments. Such filaments tend to be quite narrow and probably constitute a fairly minor influx of Agulhas water to the Atlantic, but intrusions of a much larger scale also occur from time to time (Shannon *et al.* 1990). The variation in scale, coupled with the large geographical range of filament origins, suggest there may be more than one mechanism capable of generating warm filaments in this region.

The second means, which is probably more important in terms of volume, by which Agulhas Current water is introduced into the South East Atlantic Ocean is through the shedding of warm rings at the retroflection. Lutjeharms and van Ballegooyen (1988) carried out a comprehensive study of the retroflection, using three years' thermal imagery and found a clear cyclic pattern of behaviour, consisting of a slow westward penetration (average rate of 12 cm/s) of the retroflection loop, followed by an abrupt eastward jump. In each case, where sufficient cloud free imagery was available, the jump was coincident with the shedding of a ring. On average nine rings are shed per year with no discernible seasonal pattern. In an earlier paper, Lutjeharms (1981) had shown diagrammatically the formation of a ring through pinching off of the westmost portion of the retroflection loop. Lutjeharms and van Ballegooyen (1988) surmised that ring shedding may be triggered by baroclinic instabilities in the Agulhas Return Current, giving rise to meanders which will increase the current

shear between the northern and southern parts of the loop. In each case the pinching-off action was characterised by the formation and development of a northward-extending wedge of cold Sub-Antarctic Surface Water at the Subtropical Convergence. The observed behaviour is in general agreement with numerical modelling results (Boudra and de Ruijter 1986; Boudra and Chassignet 1988; Chassignet and Boudra 1988).

Once separated, about half of the rings can be expected to travel in a northwesterly direction into the South Atlantic subtropical gyre. Gordon and Haxby (1990) used GEOSAT altimeter data to track seven such rings and concluded that about five rings can be expected to enter the South Atlantic gyre through such a route per year. The fate of the remainder is uncertain. Some may turn back into the Indian Ocean or move southwards beyond the Subtropical Convergence (Gordon and Haxby *op. cit.*). Many may remain undetected in the area southwest of South Africa. As was pointed out by Lutjeharms and Valentine (1988), these rings quickly loose their distinctive warm surface expression due to air-sea interaction (see also Gordon 1985) and thus become undetectable on thermal imagery. Lutjeharms and Valentine demonstrated several examples of rings made visible only through entrainment of warm Agulhas filaments. The drift rate for rings established by Gordon and Haxby (1990) was 5 - 8 cm/s. This is almost identical to the drift rates - 4,8 and 8,5 - reported by Olson and Evans (1986) for two rings tagged with satellite-tracked drifters in 1983. The direction of drift in this case was 299°.

Reported ring dimensions vary within fairly narrow limits ie. 370 km (Harris and van Foreest 1978), 275 km (Gordon 1985), 290-380 km (thermohaline data) and 350 km (GEOSAT data)(Gordon and Haxby 1990) and 307 km (Lutjeharms and van Ballegooyen 1988); all of which compare very well with the 342 km average diameter of the retroflection loop (Lutjeharms and van Ballegooyen *op. cit.*) and the dimension of *ca.*300 km, arrived at through numerical modelling (Chassignet and Boudra 1988).

For the two rings studied by Gordon (1985) and Olson and Evans (1986) the characteristic anticyclonic geostrophic rotation speed was 40 cm/s (maximum > 50 cm/s) for the older, more northerly, ring and 50 - 60 cm/s (maximum ≈ 90 cm/s) for the newer ring near the retroflection zone. In each case the maximum velocity was found at a distance of 110 - 130 km from the ring's centre. These velocities should be compared with the drift rates of > 100 cm/s for satellite-tracked drifters passing through the retroflection loop (Gründlingh and Lutjeharms 1979).

There is an intriguing element of consistency in the location, southwest of South Africa, where rings are most often observed. The 'anticyclonic vortex' located during a hydrographic survey in March 1969 was centred at 36,5°S : 13,5°E (Harris and van Foreest 1978), while a NOAA 5 image of 9 February 1977 showed a 'similar feature in much the same location' (Harris *et al.* 1978). The so-called 'Cape Town eddy' surveyed in November/December 1983 was centred at 36°S : 15°E (Gordon 1985). Gordon ,in fact, remarked on the similarity of the thermohaline structure and

position of this eddy to the one described by Harris and van Foreest (1978). He also pointed out that dynamic topography maps appearing in atlases revealed a topographic high in the position southwest of Cape Town, and surmised that the circulation pattern observed during the November/December 1983 cruise must represent a frequently occurring event. Lutjeharms and Valentine (1988) examined four years of METEOSAT images and found a concentration centre for rings at 36° 15'S : 18°E for which no obvious topographic or other physical explanation could be found.

In comparing the three positions given above, ie. 36,5°S : 13,5°E , 36°S : 15°E and 36° 15'S : 18°E, one notices that while the latitude position is very constant the longitudes vary considerably, which suggests a preferred line of travel rather than a concentration centre. This suggestion is not entirely in opposition to Lutjeharms and Valentine's proposal because, as they noted, their observations were based on rings made visible through entrainment of Agulhas filaments ie. they would probably not have been able to detect rings beyond some undefined westerly location. Figure 4.1, while not pretending to be an accurate presentation of bathymetry, shows a fairly pronounced westward slant of the 4000m depth contour near 35°S.A more accurate bathymetry chart (Cherkis, Fleming and Brozena 1989) in fact shows clearly a topographic ridge demarcated by the 4000 - 5000 m contours between 35 and 36°S and extending zonally westward from about 16 to 11°E. Topographic steering of the rings can therefore not be ruled out, especially since Gordon and Haxby(1990) showed that rings were deflected when crossing the

4000 m isobath of the Walvis Ridge and in light of the findings of Olson and Evans(1986) with regard to the motion of the two rings observed by them.

The implication of suggesting topographic steering is that Agulhas Rings must be of very large vertical extent. As the observed drift rates for rings are larger than theoretical values, they are probably embedded on a strong, deeper background flow which may respond to the topography rather than the ring itself (Gordon and Haxby 1990). It is in fact difficult to determine the true depth of the ring since the shear effect of the rotating feature would distort the surrounding (background) water structure - in the horizontal as well as vertical directions (McCartney and Woodgate-Jones 1991). The implications are that while the entire water column moves as a single dynamic entity, the actual depth of the entrapped water is probably much less - estimated at 670 - 1100 m by McCartney and Woodgate-Jones. Ring horizontal dimensions may for similar reasons be over estimated.

While horizontal shear may lead to erroneous estimates of ring dimensions, which in turn, may lead to over-estimates of volumes of Indian Ocean water introduced into the South Atlantic, the combined shear of a series of rings advecting northwestward past the western edge of the Agulhas Bank and southern African sub-continent, has important direct consequences for the Benguela system. This much became apparent from the November/December 1983

field work in the South East Atlantic. One of the two rings studied on this occasion was situated 300 km southwest of Cape Town and the other, more to the southeast near 39°S : 17° E (Gordon 1985). In combination the rings generated a high speed current between the rings and the continent, transporting an estimated $14_{x}10^{6}$ m^{3}/sec warm Indian Ocean surface water into the southern Benguela region (Gordon *op.cit.*). Peak velocities in this current were greater than 100 cm/s (Olson and Evans 1986). The frequency of ring shedding and the routes subsequently followed could therefore play an important - if not determining - role in water transport from the western part of the Agulhas Bank up the South African west coast. This, as was discussed in section 4.1.1, is regarded as an important factor in the recruitment cycle of the anchovy population in the southern Benguela. But, perhaps of equal or even greater importance, the process of ring shedding may provide a mechanism linking oceanographic/meteorological events in the Indian Ocean to events in the Benguela system.

4.1.3 The Antarctic Circumpolar Current (West Wind Drift)

This current, as implied by its name, is a purely wind driven flow (Deacon 1937), with easterly surface drift rates in the South East Atlantic sector ranging between 10 cm/s (35 - 40°S) and 23,3 cm/s (40 - 45°S) (Lutjeharms, Shannon and Beekman 1988). The West Wind Drift is separated from the subtropical circulation

in the South East Atlantic by the Subtropical Convergence which, according to Deacon(1937), forms solely as a result of the opposition between the subtropical and sub-Antarctic surface currents. At the Convergence, dense sub-Antarctic surface water subducts northwards beneath the subtropical surface water to form the South Atlantic Central Water (Sverdrup *et al.* 1942). This is found throughout the South East Atlantic, either as a layer separating the surface and deeper Antarctic Intermediate Water in the oceanic region, or as the sole or main water mass present over the continental shelf in the Benguela system (Shannon 1985).

Deacon (1937) gave the position of the Subtropical Convergence, in the South East Atlantic, as being between 35 and 38°S as far as 15°E, at which point it bends sharply south to about 40°S. From 70 oceanic transects through the Convergence south of Africa, Lutjeharms and Valentine (1984) arrived at an average positions of 41° 05'S and 42° 56'S for the regions 10 - 20°E and 20 - 28°E respectively which confirms the existence of a southward bend in the Convergence without defining its position. In a later paper the same authors presented diagrammatically the regional north-south limits for the Convergence (Fig.1(a) in Lutjeharms and Valentine 1988) which show the Convergence to be farthest north near 13°E ie. directly west of the Schmitt-Ott and Erica seamounts (Fig. 4.1). The north-south envelope was also widest at this point. Also, Figure 1 in Chapman, Duncombe Rae and Allanson (1987) presents the Convergence as sloping southeastwards from between these sea mounts. A permanent inflection in the Convergence, near 15°E, therefore seems

reasonably well established. The fact that this position coincides with the Agulhas Ridge and the pair of sea mounts, taken in conjunction with a similar wavelike inflection over the Agulhas Plateau further east (Fig. 4.1 ; Harris and van Foreest 1978), possibly indicates that topographic control may be involved. However, both Deacon(1937) and Lutjeharms and Valentine (1984) were of the opinion that the inflection was due to the influence of the Agulhas Current. This view would be consistent with the location of a secondary Agulhas Current retroflection point at 16°E (Lutjeharms and van Ballegooyen 1988) but is inconsistent with the more easterly position generally taken to be the main retroflection position, ie. 19°E ±1,5° (Harris, Legeckis and van Foreest 1978) or 20°E (Lutjeharms and van Ballegooyen 1988). There has been no report of an inflection in the Convergence near the latter position. This raises the possibility, that the secondary retroflection at 16°E is a consequence of the Convergence inflection and/or sea floor topography rather than the inverse. Whatever the case may be, it is clear that Agulhas Current water on occasion advances along the Convergence to positions west of this location, (eg. 9° 40'E Lutjeharms and van Ballegooyen *op. cit.*) and in an extreme case as far as 8°E (Lutjeharms 1988).

From the data obtained on transects of the Convergence between Africa and Antarctica, Lutjeharms and Valentine(1984) arrived at an average width of 225 km for the Subtropical Convergence Zone, and average temperatures at the northern and southern limits of 17,9°C and 14,2°C respectively. This gave an

average gradient of 0,047 °C/km but gradients as high as 0,15 °C/km have been found. The subsurface expression of the Convergence is generally located equatorward of the surface expression (Lutjeharms and Foldvik 1986).

The Subtropical Convergence migrates seasonally by several degrees of latitude, reaching its northernmost position in winter and most southerly extreme in summer. Using the 15° C isotherm as an indication of the northern edge of the Convergence, Gillooly and Walker (1984) arrived at a seasonal shift of 5 to 7 degrees latitude. Deacon (1937) suggested a migration of at least 6° of latitude and also suggested that the position is probably affected by meteorological changes.

Although the Subtropical Convergence is a very intense front - the most pronounced between Africa and Antarctica - there is evidence of substantial input of Sub-Antarctic Surface Water across the front into the South Atlantic Subtropical gyre. Earlier evidence for such an exchange was obtained from drift card trajectories (Shannon,Stander and Campbell 1973) and satellite-tracked drifters (Harris and Stavropoulos 1978, Lutjeharms,Shannon and Beekman 1988). Presently, filaments or wedges of cold Sub-Antarctic water extending equatorward from the Convergence, are commonly seen on satellite thermal imagery. Such intrusions seem to be mostly associated with the shedding of Agulhas rings. An event of this kind was first reported by Harris Legeckis and van Foreest (1978) from an image of 9 February 1977. More detailed descriptions of similar events were subsequently

given by Lutjeharms (1981) and Lutjeharms and van Ballegooyen (1988). The latter authors found that in almost all cases the shedding of a ring was preceded by the formation and growth of a cold wedge of Sub-Antarctic Surface water at the Subtropical Convergence. The average position, for the formation of such wedges, was found to be 18° 25'E. It is of interest to note that the positions of the few individually reported cold features are all to the west of the average location ie.

(i) 12,0°E (Lutjeharms 1988)
(ii) 16,5°E (,,)
(iii)± 17,0°E (,,)
(iv) 9,0°E (Shannon,Lutjeharms and Agenbag 1989)
(v) 15,0°E (,,)

As one possible explanation for this anomaly, it may be suggested that authors would only report the larger features and that these, for some reason, tend to originate towards the western extreme of the Agulhas Current's retroflection range. One of the two features reported by Shannon,Lutjeharms and Agenbag (*op.cit.*) was indeed of unusual extent : on 8 January 1987 the filament reached as far north as 33°S - the most northerly position yet reported. The average displacement rate for the filament (over a period of a month) was 10 - 14 cm/s, while over a 24 hour period velocities of 20 - 100 cm/s were observed. The feature had surface temperatures ranging from less than 14°C to 17°C and persisted for two months. No other work has yet been done on filaments of this nature.

4.2 CASE STUDY 1 : THE BENGUELA CURRENT REGION BETWEEN 27 AND 36^{0}S WITH AN INTRUSION OF AGULHAS WATER DURING JULY 1989.

4.2.1 Data and data processing.

Three NOAA AVHRR images were used for feature tracking purposes:

IM1 : 19 July 1989 18:33 GMT NOAA 10 Ascending pass.
IM2 : 20 July 1989 05:44 GMT NOAA 10 Descending pass.
IM3 : 20 July 1989 12:36 GMT NOAA 11 Ascending pass.

All three images were mostly cloud free, except for in the region south of 36°S - which in fact determined the southern limit in this case study. The northern limit was set by the available range of data. Seen as a single rectangle, the total image area of interest was about 1000 lines by 1000 pixels. Since an image of this size cannot be printed on a single page , data presentation is a problem. Besides, the error analysis carried out in Chapter 2 dictates that images should, where possible, be kept to areas of about 500×500 pixels. The image was therefore separated into two overlapping sections, where all parts were within 500 pixels from the coastal reference points. Transformation to Mercator projection was made using the procedure described in Chapter 2, on calibrated AVHRR band 4 data (Huh and DiRosa 1981). The sea surface temperatures in Figure 4.3 were computed using the daytime MCSST algorithm of McClain *et al.*(1985). Temperature gradient magnitudes (TGMs) were computed after transformation, using equation 3.8 and $\Delta h = 1$. Advection vectors were derived with program MTRACK applied to both the band

4 and TGM images, depending on which set seemed the most useful in a particular part of the image. MTRACK was similarly used in both 'direct' and 'window' modes, depending on circumstances.

With errors in the derived vectors expected to be inversely related to the time separation between the image pairs used for feature tracking (Chapter 2), vectors were first extracted using images IM1 and IM2, with a time interval of 11,2 hours, and which produced 533 vectors in the northern- and 250 in the southern halves of the image. However, fast currents, in excess of 100 cm/s, associated with an intrusion of warm Agulhas water, west and southwest of the Cape Peninsula, created deformations in the thermal structure and made tracking in this region very difficult. The process was therefore repeated for the southern section, using images IM2 and IM3 (time separation of 6,8 hours), resulting in a slightly larger data set of 310 vectors. In order to assess method precision, a second set of 145 vectors was derived independently for the southern section, using again IM2 and IM3. The two sets are shown in Figure 4.2 with the main set illustrated as black arrow symbols and the auxiliary test set in red; overlaps between symbols are shown in green. Eventually, 45 of the test set were incorporated into the main data set giving a total of 355 vectors for the southern image section. Both Figures 4.4 (southern half) and 4.5 (northern half), actually contain a slightly smaller number of vectors than the numbers quoted above, due to some parts of the original geometrically corrected areas having been trimmed off to produce rectangular images.

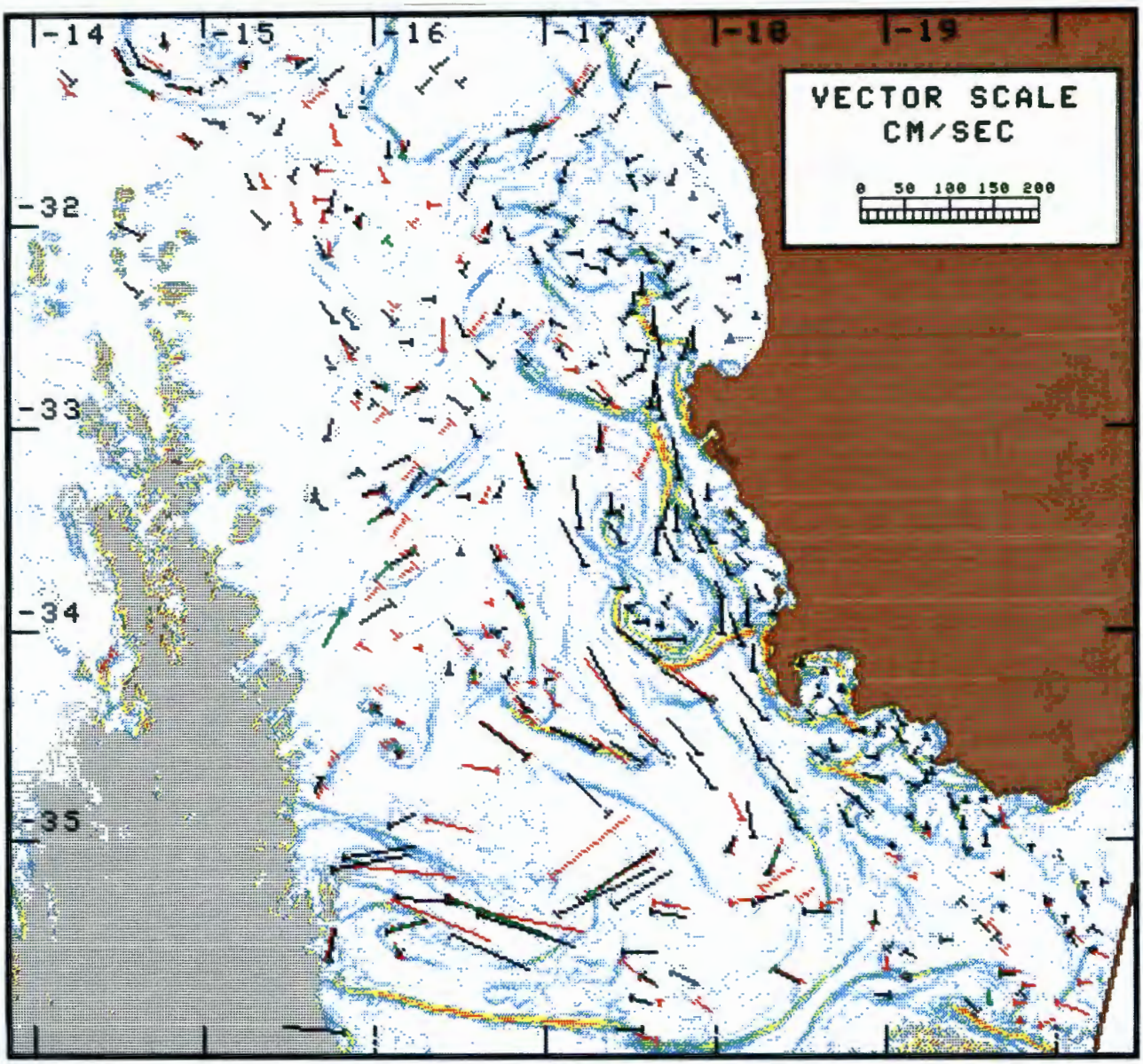


Fig. 4.2 A temperature gradient magnitude image derived from a NOAA-10 image of 20 July 1989 (the weakest gradients are shown in blue and the strongest in orange).The two sets of advection vectors derived to test vector computation precision are superimposed in the gradient structure (black = main set, red = auxiliary set, green indicates overlap between the two sets). Vectors are indicated as arrow symbols, each with a feather on the tail.

In addition to the digital image data used for feature tracking (IM1,IM2 and IM3), use was made of the Institute's photographic archive of AVHRR images. The archive consists of 25x25 cm negatives of contrast enhanced raw AVHRR band 4 data - no calibration or geometric correction was applied. The images are acquired at a rate of one to two per week and serve as an inexpensive data source for monitoring events in the South East Atlantic and South West Indian Ocean. In this study the archive imagery proved invaluable as a means of tracing the development histories of some of the more prominent features in the case study imagery. Since the archive imagery contain no latitude/longitude grid, the positions of features or their dimensions were determined by overlaying one archive image with another corresponding to one of the digital images used in the study. Since the digital images were geometrically corrected, the second image provided a scale for converting relative measures on the negatives to absolute units.

A hake recruitment cruise during July 1989 (Cruise HK 075) provided some hydrographic data. Hydrographic sampling had however been done at randomly spaced trawl positions- usually close inshore on the inner shelf - and was generally only of limited value to this case study.

4.2.2 Discussion :

1. Vector precision.

Figure 4.2 illustrates the previously mentioned problem encountered when attempting to do a quantitative comparison of

two vector sets in a field with high spatial variability in both speed and direction. Unless two vectors have virtually the same origin, it becomes hard to tell errors from real spatial variability. Therefore, in order to obtain an estimate of the degree of variance between the two sets, a subset of 63 vector pairs, judged to be close enough in space to be comparable, were extracted from the primary sets. Vectors in the subset ranged in size from 6 to 140 cm/s with an average of 32 cm/s. For each vector pair the size (speed) of the auxiliary vector was subtracted from that of the main set vector and the same for the directions. This gave an average size difference of 1,3 cm/s and an average directional difference of 5°, ie. the main set vectors were slightly larger and rotated marginally more clockwise than the vectors in the auxiliary set. However, when the root-mean-square (RMS) differences are computed, considerable larger values are obtained, ie. 8,2 cm/s and 31°. The relatively small averages and large RMSs indicate a fairly even distribution of positive and negative differences and hence the absence of significant bias.

With the time separation of 6,8 hours between the two images and the Mercator projection scale used (one minute of longitude equal to one pixel), the smallest velocity increment which can be properly resolved is about 6 cm/s. The RMS error of 8,2 cm/s therefore corresponds to an error of about one pixel. At very low velocities, directional resolution becomes very poor - 90° in the extreme case! Examples of this are seen in the low velocity

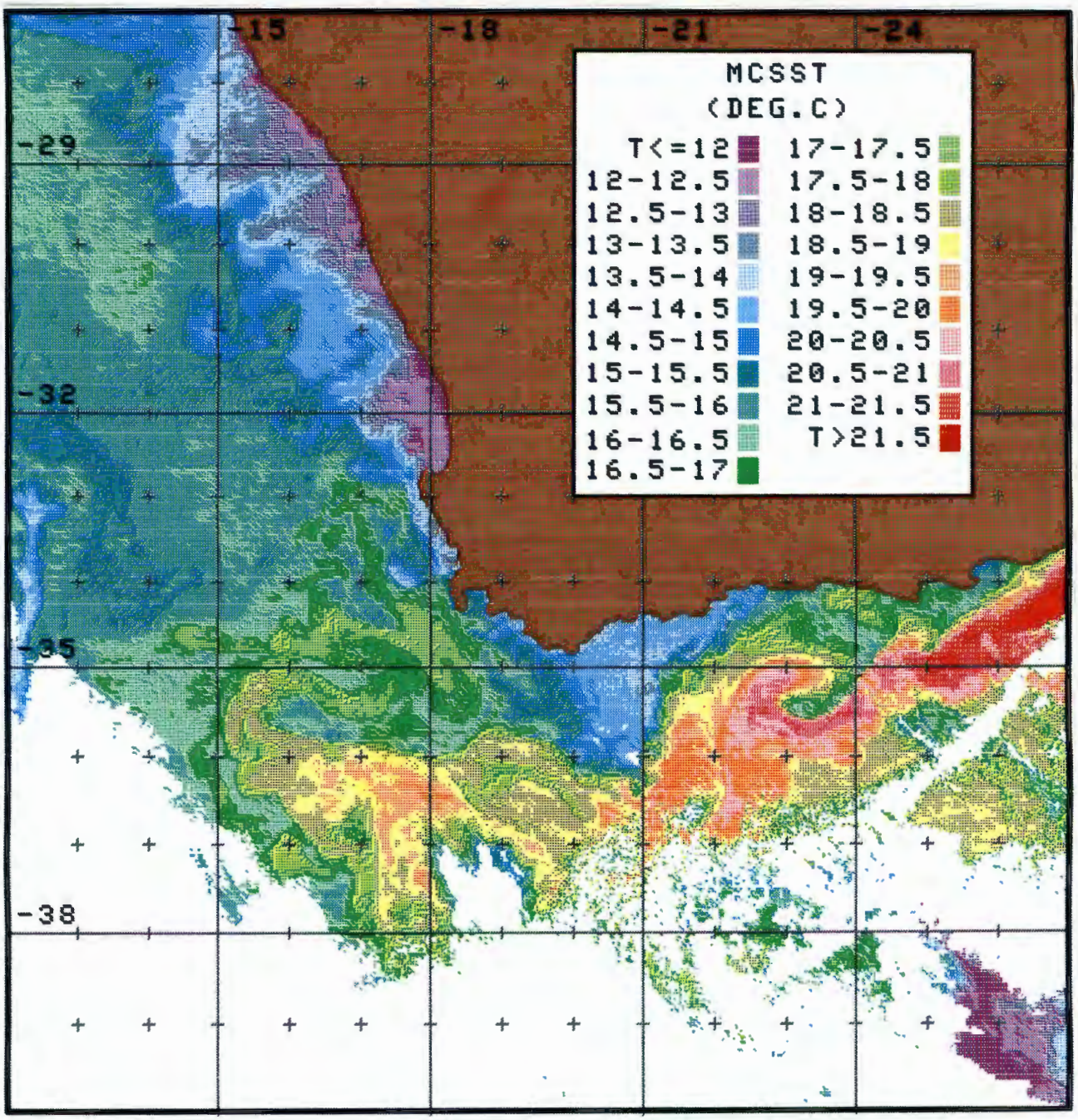


Fig. 4.3 The sea surface temperature distribution in the case study region as obtained from a NOAA-11 AVHRR image on 20 July 1989, 12:36 GMT.

region on the Agulhas Bank near 35°S : 19°E and also near 35° 20'S : 19° 40'E. The large RMS directional error is derived from this source, so that by removing six of the smallest vector pairs from the test set, the RMS error reduces to a very reasonable 12,4°.

The part of the image demonstrating the largest degree of inconsistency between the two data sets is in a low gradient region near 32°S : 16°E. In this position the auxiliary test set shows a more pronounced degree of cyclonic rotation than the main set. The overall effect is quite small however and does not detract from the general impression that one could describe the flow field in this image on hand of either set and arrive at virtually identical conclusions in terms of speed and direction ie. the results seem to be reproducible with good precision.

2. Oceanography : The southern section as portrayed by Figure 4.4

The warm water intrusion:

The sea surface temperature (SST) distribution and advection field in the southern half of the image (Fig. 4.4) is dominated by an intrusion of warm water. One of the HK075 stations (Station DT001 of 22/7/1989) was located in the warm water at 34° 37,4'S : 17° 51,4'E , just off the shelf edge in 896m of water (station position marked in Figure 4.4). Surface measurements (2m) at the station indicated : T = 17,69°C , S = 35,61 $\times 10^{-3}$ and σ_t = 25,8168. AVHRR SST at the station position was 17,6°C. From

average TS profiles (Fig.11 in Nelson (1985)) the salinity, corresponding to the temperature at the station, should be 35,4 - 35,5. Also, Shannon (1966) reported a surface temperature of 15,9°C and salinity of 35,4 $\times 10^{-3}$ near this position for July 1960. It therefor seems that the salinity measured at station DT001 was rather higher than would normally be expected for South Atlantic surface water of that temperature. The high salinity tentatively identifies the warm water as being of Agulhas Current origin, but as discussed in Section 4.1.1, discrimination between Agulhas Current water and South Atlantic subtropical water, on the basis of thermohaline characteristics, is difficult.

For the same reason, determination of the depth of the feature was very difficult. Shannon (1985) states that such warm filaments are generally less than 50m deep, and Lutjeharms and Stockton (1987) also suggest that they do not exceed depths of 50 - 100m. Station DT001 temperature and salinity profiles agree fairly well with Nelson's (1985) diagram, at depths of 150m and deeper - although salinities still remain slightly higher than Nelson's but less than that of Sverdrup *et al.*'s (1942) salinities for South Atlantic Central Water (SACW). From 600m and deeper DT001 salinities merge with the SACW characteristics. Comparison with the July 1960 profiles (Shannon 1966), show DT001 temperatures higher than Shannon's but becoming similar from 700m onwards. Shannon did not report subsurface salinities at this particular location, but 100m horizontal sections from April and

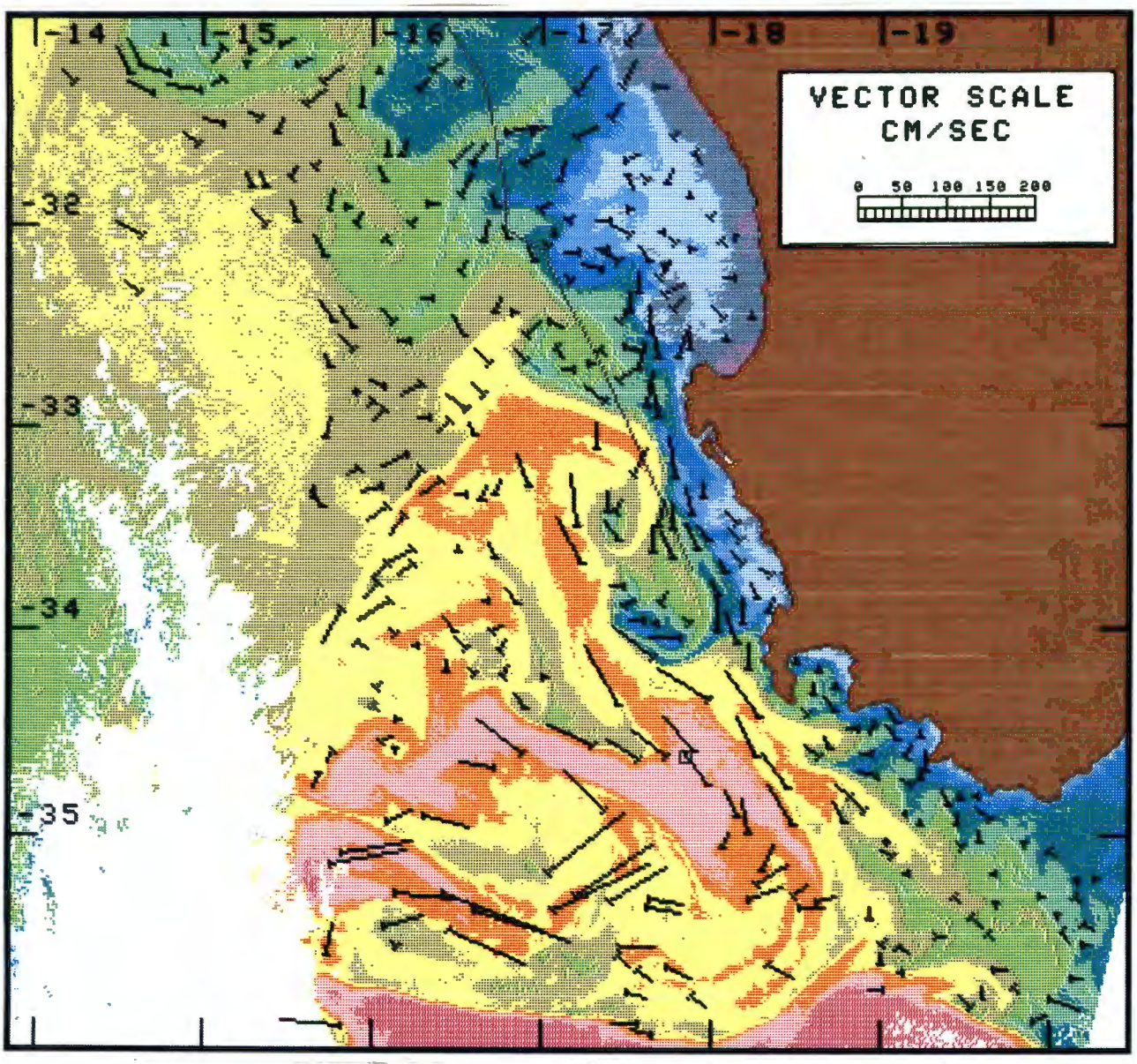


Fig. 4.4 The surface advection pattern for the southern half of the case study area. Advection vectors are shown as arrow symbols, each with a feather on the tail. The image shows the SST pattern from AVHRR band 4 of NOAA-10 20 July 1989 (low SSTs are indicated by blue and high SSTs by red/pink). The position of the hydrographic station DT001 is shown, as well as the drogue track from Nelson and Hutchings (1983).

October 1960, indicate salinities of 35,1 and 35,4 compared with 35,51 at station DT001 in July 1989. From this one would deduce that the most probable depth for the warm water feature was 100 - 150m but that it could also have been much deeper. The fact that the feature was located directly seaward of the shelf edge suggests topographical steering and hence possibly a deep feature, but the arguments applied to the drift of Agulhas rings (Section 4.1.2), ie. dynamic linkage of a shallow feature with a deeper flow - perhaps a shelf-edge jet such as exists further north off the Cape Peninsula (Bang and Andrews 1974) - are valid in this case and present a scenario more in line with the available information.

Although the intruding warm water cannot be conclusively identified as of Agulhas Current origin through use of the available thermohaline data alone, Figure 4.3 shows quite clearly that the filament which dominates the scene in Figure 4.4, is only a relatively small extension of a much larger, convoluted mass of warm water which is apparently directly linked with the Agulhas Current retroflection. The retroflection itself is mostly hidden by cloud, but seems to be located near 38°S : 20°E. Warm water, with a maximum temperature of about 20°C, emanates as a broad plume from a position west of this position and stretches about 200 km in a northwestward direction to latitude 36°S. At this point the plume curves westward and is joined by a secondary flow of water from the southern tip of the Agulhas Bank (37°S : 21°E).

From the thermal structure in Figure 4.3, the impression is gained that warm water flows zonally westward along the 36°S parallel, from about 20°E to 16°E. This impression is confirmed by the vectors shown at the bottom of Figure 4.4. The velocities shown in this position range between 25 and 74 cm/s, but two other vectors (lost when Figure 4.4 was trimmed to a rectangular shape) suggest a steep southward velocity gradient : (i) 92 cm/s, 267° at 36° 4'S : 16° 43'E and (ii) 115 °, 283° at 36° 8'S : 17° 24'E. These velocities are similar to those previously measured along the rims of Agulhas rings (Olson and Evans 1986), but the thermal structure in Figure 4.3 bears little visual relation to a ring. It is however interesting to note the coincidence of the northern limit of the warm features seen in Figure 4.3 and the east/west sea floor topographic structure situated between 35 and 36°S, previously discussed as a possible mechanism controlling the tracks of Agulhas rings (Section 4.1.2).

The intense cyclonic/anticyclonic flow pattern seen in the bottom centre of Figure 4.4 has associated with it very high velocities, typically 70 cm/s, but with peak values in excess of 100 cm/s (maximum = 140 cm/s) in the vicinity of 35° 20'S : 17°E, where the flow direction changes from cyclonic to anticyclonic. This pattern developed in the period between 7 July (for which a moderately clear NOAA-11 archive photo-negative image is available) and the 19 July case study imagery.On the 7 July image a warm filament could be seen to issue from the main body of warm water, in virtually the same position as in Figure 4.4, ie. near 34° 30'S : 17° E, but instead of being 'folded' as in figure 4.4,

described a roughly semicircular, cyclonic trajectory. No usable imagery were received between these two dates , on account of the cloud cover associated with the passage of a pair of intense atmospheric cyclones. Geostrophic wind speeds of up to 65 km/h were estimated from the pressure gradients near 35°S : 15°E (Daily Weather Bulletin, South African Weather Bureau) during this period and were possibly instrumental in the creation of the flow pattern as observed in Figure 4.4.

Advection patterns along the West Coast.

The highest velocities along the West Coast are associated with jet of warm water emanating from the folded and convoluted mass in the bottom part of Figure 4.4. The jet is seen to flow parallel to the coast between 35° 10'S : 18° 30'E and 32° 48'S : 16° 45'E. In the south it is in excess of 40 km wide, but narrows to less than 20 km off the Cape Peninsula and finally terminates in a vortex dipole structure west of Cape Columbine. The core position - taken as the centre of the warmest water shown in Figure 4.4 - wanders over depths ranging from 1500m, in the south, to 400m, off Columbine, but is generally positioned just seaward of the shelf edge. The sharp SST gradient seen in the south (shown as a rapid transition from the red to yellow in Figure 4.4) coincides almost precisely with the shelf edge. Further north the edge of the feature becomes less well defined, but the position relative to the shelf edge seems to be maintained. Core velocities vary from 45 cm/s, in the southern-most position, to a maximum of 88 cm/s, at the point off the Cape

Peninsula where the flow becomes horizontally compressed; it then decelerates again to about 40 cm/s west-southwest of Columbine. Inshore of the jet, velocities generally decrease rapidly to about 15 - 20 cm/s, but maintain more or less the same flow direction as the jet. This pattern is interrupted by the cyclonic rotation associated with two cold core eddies. The southernmost of the two, west of the Cape Peninsula, seems to give rise to a divergence, resulting in two parallel longshore flows, the first being the warm jet and the second, a flow of colder water inshore. The highest velocities associated with the inshore branch occur along the thermal front, located between 20 and 40 km offshore, and are typically 32 - 34 cm/s. Local acceleration to 40 - 44 cm/s occurs at the point where the flow passes between the Cape Peninsula and the cold eddy, and again in the equivalent position off Cape Columbine, reaching a maximum of 51 cm/s in the latter position. North of Cape Columbine (32° 10'S : 17° 40'E) the flow turns abruptly offshore along the edge of a small plume of cold water located there. There is a good agreement between this frontal flow and the path of a satellite-tracked drogue released on 23 February 1979 (Nelson and Hutchings 1983) shown superimposed on Figure 4.4. Shoreward of the thermal front, SSTs decrease from about 16,6°C (in the warm jet) to about 13 - 14°C in the nearshore region between the Cape Peninsula and Columbine; velocities decrease at the same time from 32 - 34 cm/s to 14 - 20 cm/s.

The existence of a fast shelf-edge jet off the Cape Peninsula is well established (Section 4.1.1). Bang and Andrews (1974)

recorded an equatorward surface flow of 60 - 80 cm/s and proposed that the jet transported Agulhas Current water northwards along the West Coast. Velocities in the warm filament off the Peninsula (according to Fig. 4.4) are between 63 and 88 cm/s and therefore are in very good agreement with Bang and Andrews. These authors further described strong anticyclonic circulation to the west of the jet with poleward velocities of 40 - 60 cm/s from about 140 km offshore and suggested this to be a form of compensation for transport in the jet. Figure 4.4 show such poleward flow originating at the tip of the warm filament off Columbine. Velocities in this current increase from about 32 cm/s, at Columbine, to a maximum of 51 cm/s, on latitude 34°S. Which is once again similar to the findings of Bang and Andrews, but the poleward flow is considerably further offshore, ie. about 220 km. The weak equatorward flow of 20 cm/s found nearshore (15 km) by Bang and Andrews is similar to the 14 - 20 cm/s velocities found inshore of the thermal front during this study. However Figure 4.3 provides no indication of the near shore poleward flow reported by several authors. Bang and Andrews, for example, described a narrow - approximately 20 km wide - poleward current of 20 cm/s, inshore and adjacent to the thermal front. Poleward motion was also found by Shannon (1966),Duncan and Nell (1969) and Nelson (1985). The flow is barotropic and subject to reversals.

There is no mention in the literature of a divergence off the Cape Peninsula, although the divergent flow seen in Figure 4.4 is similar, in appearance, to the flow pattern off Columbine -

a location of known divergence (Shannon 1985). In both cases the divergence seems to be related to the presence of a plume of cold water, terminating in a cyclonic eddy. The divergence at Columbine was described by Shannon (*op.cit.*) as consisting of two fonts and two jets : (i) a front near the 200m-isobath with a shallow, time variable jet in which both equatorward and poleward velocities of about 60 cm/s have been recorded and,(ii) the main oceanic front near the 400m-isobath, with associated equatorward jet of an estimated velocity also equal to about 60 cm/s. In Figure 4.4, the core of the nearshore equatorward flow is located about 20 km off Cape Columbine at a water depth of 200 m and a velocity of 45 cm/s. It is a well defined jet, corresponding with a reasonably well defined front and thus agrees closely with Shannon's description. There are clear signs of a divergence point near 32° 50'S : 17° 34'E which coincides with Shannon's divergence point, and is in fact the contact point between the warm water in the cyclonic vortex and the nearshore cold water. The intense thermal front between these waters stretches in a northwestward direction and curve northward near 32° 49'S : 16° 53'E. The flow along the front probably represents the westerly branch of Shannon's divergence. Velocities along the front range between 18 and 32 cm/s, ie. much weaker than the 60 cm/s estimated by Shannon. However, Figure 4.4 represents quiescent winter upwelling conditions and one may expect both a more extensive upwelling plume and higher velocities during active summer upwelling.

3. Oceanography : In the northern section as portrayed by Figure 4.5

Figure 4.4 provides evidence for the existence of two longshore currents in the Cape Peninsula to Cape Columbine region and these were discussed in the preceding section. The current nearer the coast originates at the divergence off the Cape Peninsula and flows first as a fast linear jet along the main thermal front as far as Cape Columbine. From this point it starts to meander in sympathy with the irregular front created by the cold plumes protruding from the nearshore region. The buoy track superimposed on Figures 4.4 and 4.5 compares fairly well with the course of this current. The second current is positioned further from the coast and transports warm water of Agulhas Current origin, from a point south of the Cape Peninsula to as far as Cape Columbine, from where it flows along an almost straight northwesterly line on the outskirts of the coastal cold water plumes. The two streams come in contact between 29 and 30°S and perhaps again near the northeastern corner of Figure 4.5. The drogue track shown in Figures 4.4 and 4.5 agrees closely with that of another drogue released on 8 March 1977 at about the same position (Harris and Shannon 1979, Nelson and Hutchings 1983). Although it is clear from the vector data presented that the drogues would probably have followed a different track if released further offshore, Nelson and Hutchings' supposition that this particular track represents the core of the Benguela Current, is a sensible one from the point of view that the track traversed the coastal upwelling zone which is regarded as the classical characteristic

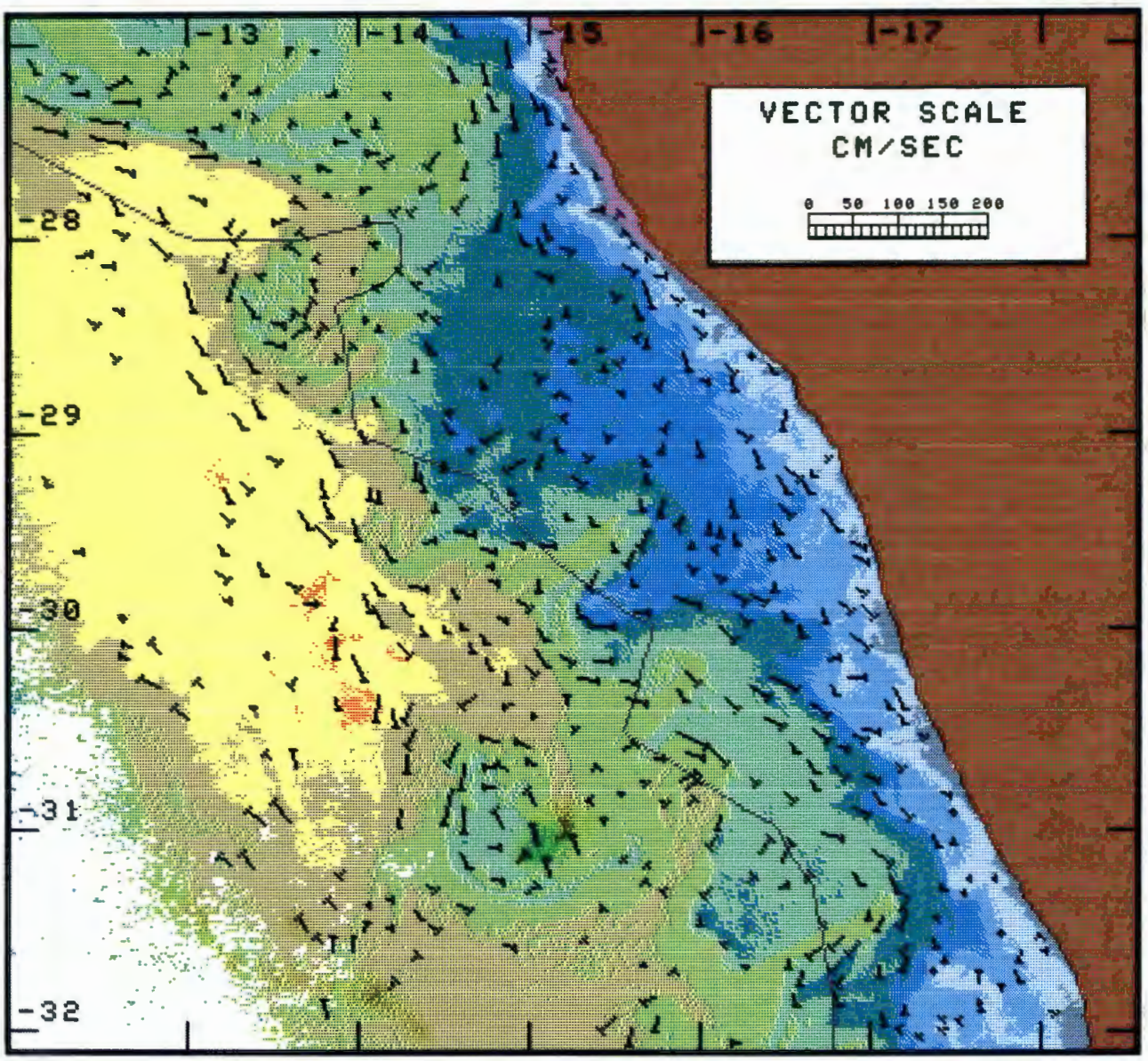


Fig. 4.5 The surface advection pattern for the northern part of the case study area. Advection vectors are shown as arrow symbols, each with a feather on the tail. The image shows the SST distribution from AVHRR band 4 of NOAA-10 on 19 July 1989 (Low SSTs are indicated in blue and high SSTs in yellow/orange - the same colour code as used for Figure 4.4

of the Benguela.

It is not easy to reconcile the flow pattern portrayed by Figures 4.4 and 4.5 with the general perception of a northwestward flowing Benguela Current. More or less uninterrupted northwesterly flow is only seen along the outermost of the two streams described above. Flow velocities vary considerably. High velocities (up to 88 cm/s) occur in the south near the Cape Peninsula and again near the northwest corner of Figure 4.5 (up to 50 cm/s). In between these two positions, speeds are less variable and mostly within a range of 20 to 30 cm/s, which is in fair agreement with the 24 cm/s mean velocity obtained by Harris and Shannon (1979) and 25 cm/s indicated by Shannon(1985) as typical for the southern Benguela. In the adjacent nearshore zone of cool upwelled water, the flow field is modulated on mesoscale by tongues and plumes of cool water protruding from the coast. There is a clear tendency for advection to take place parallel to the isotherms in these features, giving rise to offshore transport along the poleward sides and onshore transport along the equatorward sides (eg. at 30°S and between 15 and 16°E in Fig 4.5).

Virtually all of the cold water plumes seen in Figures 4.4 and 4.5, terminate in a partially detached cyclonic eddy. Six such eddies can be seen between the Cape Peninsula and latitude 28°S and possibly a seventh in the northwestern corner of Figure 4.5 (the cold cores in the vortex dipole off Columbine are of different origin and not counted). The southernmost two eddies

are positioned in 300m of water, on the shelf but close to the shelf edge, at 63 and 72 km offshore respectively. However, those to the north of Columbine are positioned off the shelf, and progressively further offshore and in correspondingly deeper water till the most northerly one, in the top left corner of Figure 4.5, is 290 km offshore and in 3800m of water. Since the most coherent surface flow in Figures 4.4 and 4.5 occurs at a distance from the coast corresponding with the outer limit of these eddies, the implication is that the more coherent northwestward flow is also found at progressively greater distances from the coast.

The seven eddies occur at fairly regularly spaced intervals. From south to north the spacing between adjacent eddies are : 153, 133, 147, 194, 136 and 174 km (mean = 156 km, standard deviation = 24 km). Judging by the thermal structures, all these eddies are cyclonic, an impression which is confirmed in a few instances by the sense of the derived vectors - particularly clear in the case of the two eddies centred at 31°S and 28° 18'S. The derived velocities are not symmetrical about the vortices, but are larger along the westerly or southwesterly edges. In the case of the eddy at 31°S, the maximum velocity along the western edge is 45 cm/s and in the case of the other, 38 cm/s - hence representing a local acceleration in the general flow field. The observed asymmetry could be the consequence of either vector addition between the background flow and the rotating structure drifting with it or shear induced local acceleration produced by the background acting on the 'exposed' westerly/southwesterly edge

of a stationary structure. Overlaying the image of 20 July (from which Figures 4.4 and 4.5 were derived) with an image of 28 July (archive photo-negative, not illustrated), shows that, with the exception of the Columbine eddy, the other six can easily be recognised on both images and remained in identical or very similar positions during that period. The eddies were therefore not drifting with the background flow and hence it can be concluded that the asymmetry is due to shear effects. This conclusion is substantiated by the tendency towards elongation of the eddy in the direction of the background flow - eg. in the case of the eddy off the Cape Peninsula, the structure has a long axis of about 41 km and a short axis of 24 km.

4. The relevance to fish populations.

The relevance of the surface advection pattern to the spawning/recruitment cycle of the anchovies was discussed in sections 4.1.1 and 4.1.2. Spawning takes place primarily on the Agulhas Bank, which is thought to contain insufficient planktonic material to support the development of the young fish, hence the need for the eggs and larvae to migrate or be transported to the West Coast upwelling region with its higher biological production rates. One suspects that the spawning cycle evolved to be in tune with the average ambient current pattern, but that it may be disturbed by short term variations such as perhaps the intrusion of the Agulhas Current filament observed in this study. The detailed synoptic description of the surface advection pattern

obtained during this investigation therefore seems to lend itself very well to the study of such physical/biological interactions. It is however necessary to point out that this case study is based upon mid-winter images, while peak anchovy spawning occurs during spring and summer. the following discussion of observed advection features, in terms of their influence on the transportation of eggs and larvae, is therefore not set within the correct seasonal context. However, since no spring/summer advection pattern is available at the moment, we chose to ignore the seasonal mismatch for the time being, in the believe that some insight into the influence of the physical environment on the anchovy population may nonetheless be gained.

Figures 4.4 and 4.5 provide ample evidence of favourable transportation mechanisms from the northwestern part of the Bank. It is however also clear that, under conditions such as existed during this case study, spawn products from the more southerly section of the Bank, were at risk of being swept offshore through entrainment into the warm water filament present along the southwestern edge of the Bank. It is not easy the access the actual risk since not many vectors were obtained in that particular region due to cloud cover. The few available vectors are on the edge of the feature and show westward flow of 20 - 25 cm/s. It is likely that velocities towards the core of the warm feature were higher as was demonstrated by the velocity gradient further west. There is however, on the basis of the available advection data in Figure 4.4, not much evidence of major entrainment of Agulhas Bank water. In fact, the rather confused

pattern directly to the north of the warm water suggests intrusion of warm water on to the Bank, rather than removal of water. The exception being a small cyclonic curving plume - presumably a shear effect - located near 35° 40'S : 19° 30'E, on the edge of the warm feature, which may be expected to entrain some Bank water.

In contrast, clear evidence of entrainment is seen in the northwestern part of the Agulhas Bank, just south and southeast of the Cape Peninsula. Water entrained at this point could be either injected directly into the cold water regime or be transported as part of the warm filament along the seaward edge of the cold water. Eggs produced in the nearshore region between the Cape Peninsula and Cape Agulhas are most likely to follow the first route and are least likely to suffer losses through advective processes. Most of this material would probably be transported along the frontal jet into St Helena Bay, north of Cape Columbine - a distance of about 220 km. At an average rate of about 35 cm/s this journey would be completed in seven or eight days. A portion may enter Table Bay, north of the Cape Peninsula, within about three days while part may be transported past Columbine, where onshore transport is available on the equatorward side of each of the various cold plumes, particularly so, near 31°S, 29°S and 27°S.

Eggs produced in the same part of the Bank but further from the coast, are more likely to enter into the jet located on the outside of the frontal region. The risk of advective losses from

this route is seen to be substantially higher than from the more inshore route. Offshore advection occurs mainly in three locations, ie. near 33°S (reached after 6 - 7 days), near 31°S (reached after about 18 days) and near 27° 30'S (reached after about 37 days). Even if it is assumed that the longshore transportation rate during active summer upwelling may be less than derived from the mid-winter conditions portrayed by this case study, it still seems possible for anchovy larvae to reach the Luderitz upwelling centre near 27°S (see Section 4.1.1) in less than ten weeks, at which stage the cruising speed for the young fish would only be 2 - 4 cm/s (Nelson and Hutchings 1987) and much less than the ambient offshore flow observed at several points (Fig. 4.5). Onshore transport from the outer longshore route, as for the nearshore route, would be facilitated by the cross shelf flow associated with the cold water plumes. This should lead to an uneven distribution of larvae in the nearshore region and the locations of highest concentration should be roughly predictable from the positions of quasi-stationary upwelling plumes/filaments. The advection patterns in Figures 4.4 and 4.5 however only indicate two coastal positions where the flow is southerly or sufficiently weak northerly, to allow retention of juvenile fish. The first being the region from Cape Columbine northwards to 31°S, and to a lesser extent, the Orange River Bight, near 29°S. Seen in combination with the onshore transportation pattern, one would expect the highest concentrations of young fish to occur in these two areas and there is some existing evidence to support this hypothesis (eg. Cruickshank, Hampton and Armstrong 1990). Presumably the

stationary cyclonic eddies, on the seaward side of the upwelling plumes, could also act as retaining centres for larvae but there is no supporting biological evidence for this.

4.3 CASE STUDY 2 : THE SOUTH EAST ATLANTIC OCEAN DURING AN INTRUSION OF SUBTROPICAL CONVERGENCE WATER. MAY 1990.

4.3.1 Data and data processing :

Three NOAA AVHRR images were used for feature tracking purposes:

IM1 : 13 May 1990 13:43 GMT. NOAA 11. Ascending pass.
IM2 : 14 May 1990 05:44 GMT. NOAA 11. Descending pass.
IM3 : 14 May 1990 13:33 GMT. NOAA 11. Ascending pass.

The area of interest is shown by Figure 4.6 and its boundaries were determined largely by the cloud free portions of the available imagery. IM3, used for illustration purposes (Fig. 4.6, 4.7 and 4.8) was the most cloud free. Different parts of the other images were cloudy and necessitated the extraction of two sets of vectors. The main set, consisting of 392 vectors, was derived from IM2 and IM3 which were separated in time by about 12,4 hours; vectors in this set are indicated by black arrow symbols. A second set of 80 vectors, was obtained from IM1 and IM3 which were separated by about 23,8 hours; vectors in this set are shown as red arrows. The second set was derived primarily as a means of filling out the main data set in cloudy areas, but the data also serve as an additional check on data precision. Vector extraction was done with MTRACK in a manner similar to, and following the image processing steps, described for Case Study

1 (Section 4.2). However the sub-image transformed to Mercator was very large : approximately 900 lines by 1300 pixels. Hence, in compliance with the findings of Chapter 2, the transformation was performed on three partially overlapping sections, neither of which was more than 550 pixels wide. For illustration purposes the composite obtained after geometric correction, had to be reproduced as two separate, overlapping western and eastern sections (Figs 4.7 and 4.8). The sea surface temperatures in Fig.4.6 were computed using the day-time MCSST algorithm of McClain *et al.*(1985). As in case study 1, use was made of archive imagery to trace the development histories of some of the features.

4.3.2 Discussion.

General

The image of 14 May, shown in Figure 4.6, portrays a complex and apparently dynamic oceanographic scenario, composed of several large scale features. The most prominent of these features being the tall filament of cold water, originating from a wedge on the Subtropical Convergence (STC), near 38°S : 13°E and stretching northward between longitudes 13 and 14°E to terminate in a large cold core gyre centred on about 33° 40'S : 14°E. This feature is also entraining a filament of cool water flowing zonally westward along about 35°S from the western edge of the Agulhas Bank.

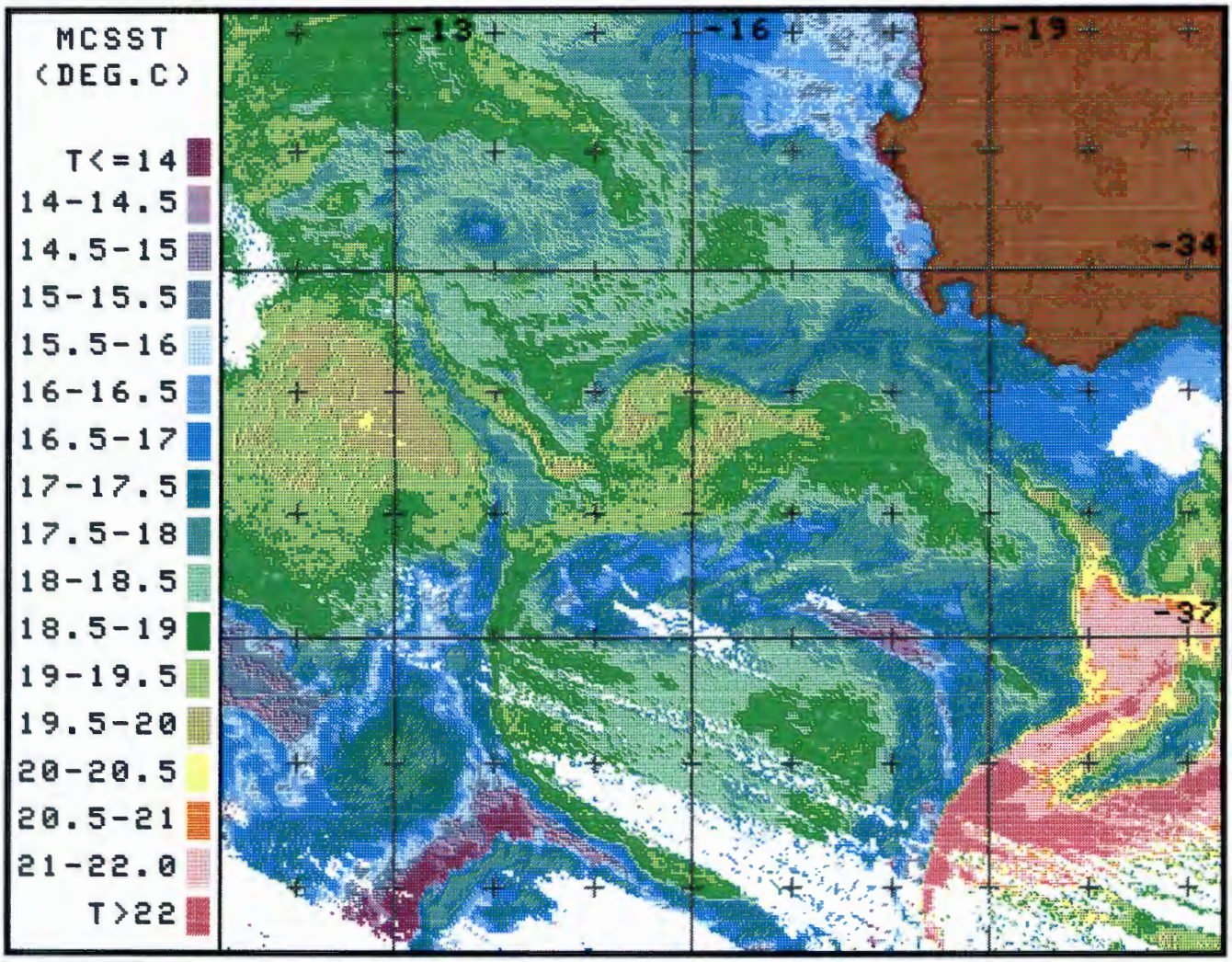


Fig. 4.6 The sea surface temperature structure in the case study area as obtained from a NOAA-11 AVHRR image on 14 May 1990.

Judging by the SST pattern in Figure 4.6, this filament seems to be composed mainly of western Agulhas Bank water with a contribution of warm water, originating at the Agulhas Retroflection (39°S : 19°E). Inspection of the vector pattern in Figure 4.8, however, also shows a major contribution of Sub-Antarctic Surface Water via a second filament originating on the STC, directly west of the retroflection. From its origin, this filament describes a curiously contorted path which is clearly

revealed by the current vectors. At first the flow is almost directly northwards in the form of a thin cold filament to 37°S : 18° 30'E, upon which it turns abruptly westward for a distance of about 300 km to 36° 30'S : 15° 30'E, where it reverses it's course, flowing eastward and meets up with the Agulhas Current filament at a point near 36° 30'S : 19° 30'E. This filament will be referred to as STCF2 to distinguish it from the first mentioned Subtropical Convergence filament which will be labelled STCF1; the cold core gyre at the tip of STCF1 will be referred to as the 'STCF1-eddy'. Furthermore the warm filament will be referred to as the 'Retroflection-filament' and the cold westward flow, originating on the western Agulhas Bank, as the 'Bank-filament'. In addition to the features already mentioned we shall also present evidence which will lead to the identification of two Agulhas Current rings, which will be labelled 'RING1' and 'RING2'. All of the major features may be located on Figure 4.9 which was drawn after the thermal structure in Figure 4.6.

As an unfortunate consequence of the generally chronic cloudiness of this region the last reasonably clear image received prior to IM1 (13 May) was from NOAA 11 on the 18th of March.At first glance this was thought to be too large a time gap to be of much use in following the development history of the observed features, but when the two images were compared (actually the 18 March image was compared with IM3 ie. the 14 May image), a surprising amount could be learned. In the following sections we shall discuss aspects of the development and advection characteristics of the various features mentioned in the previous

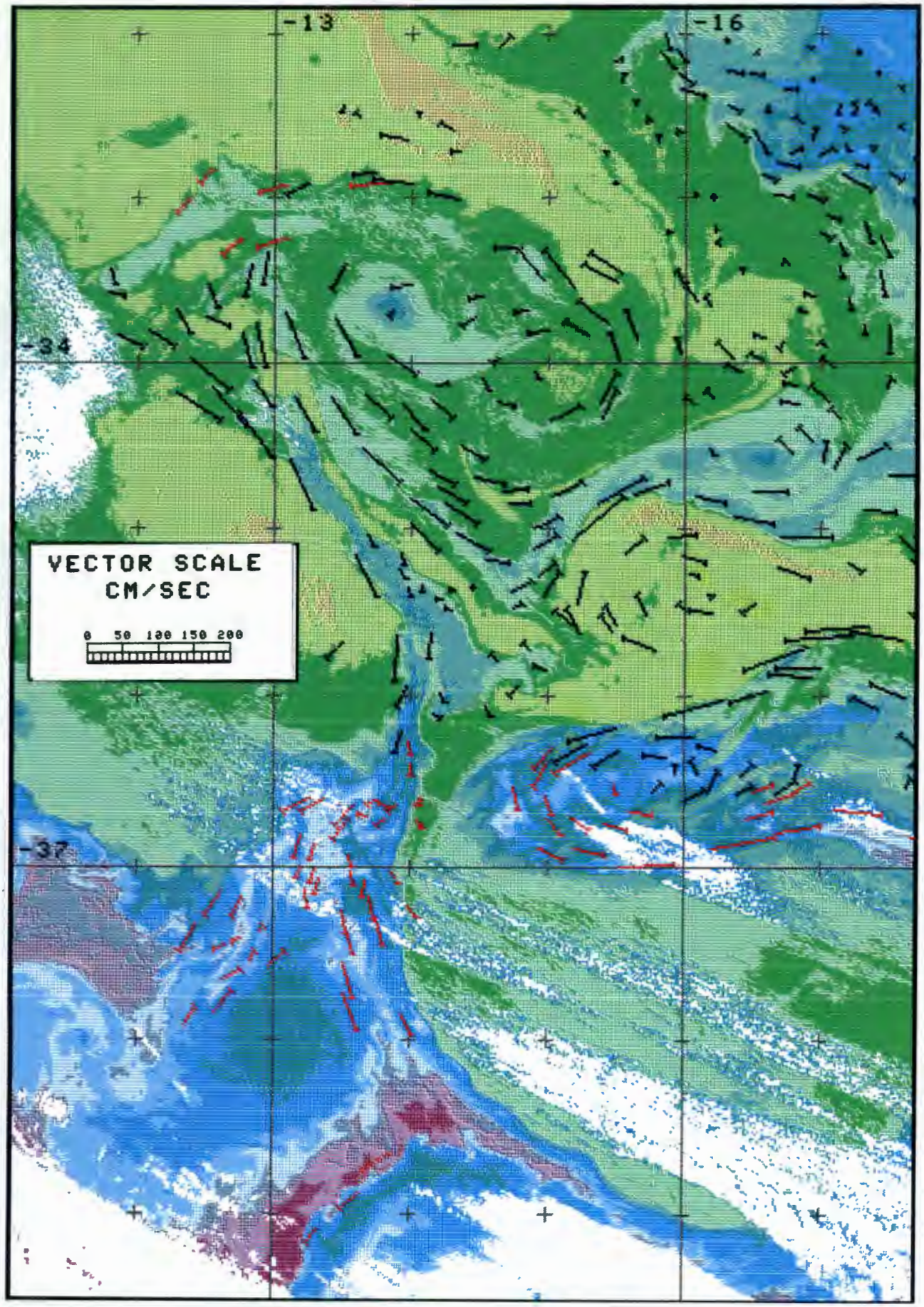


Fig. 4.7 The surface advection pattern in the western half of the study area. The black arrows were obtained from images IM2 and IM3 and the red arrows from images IM1 and IM3. The image is AVHRR band 4 of NOAA-11 on 14 May 1990.

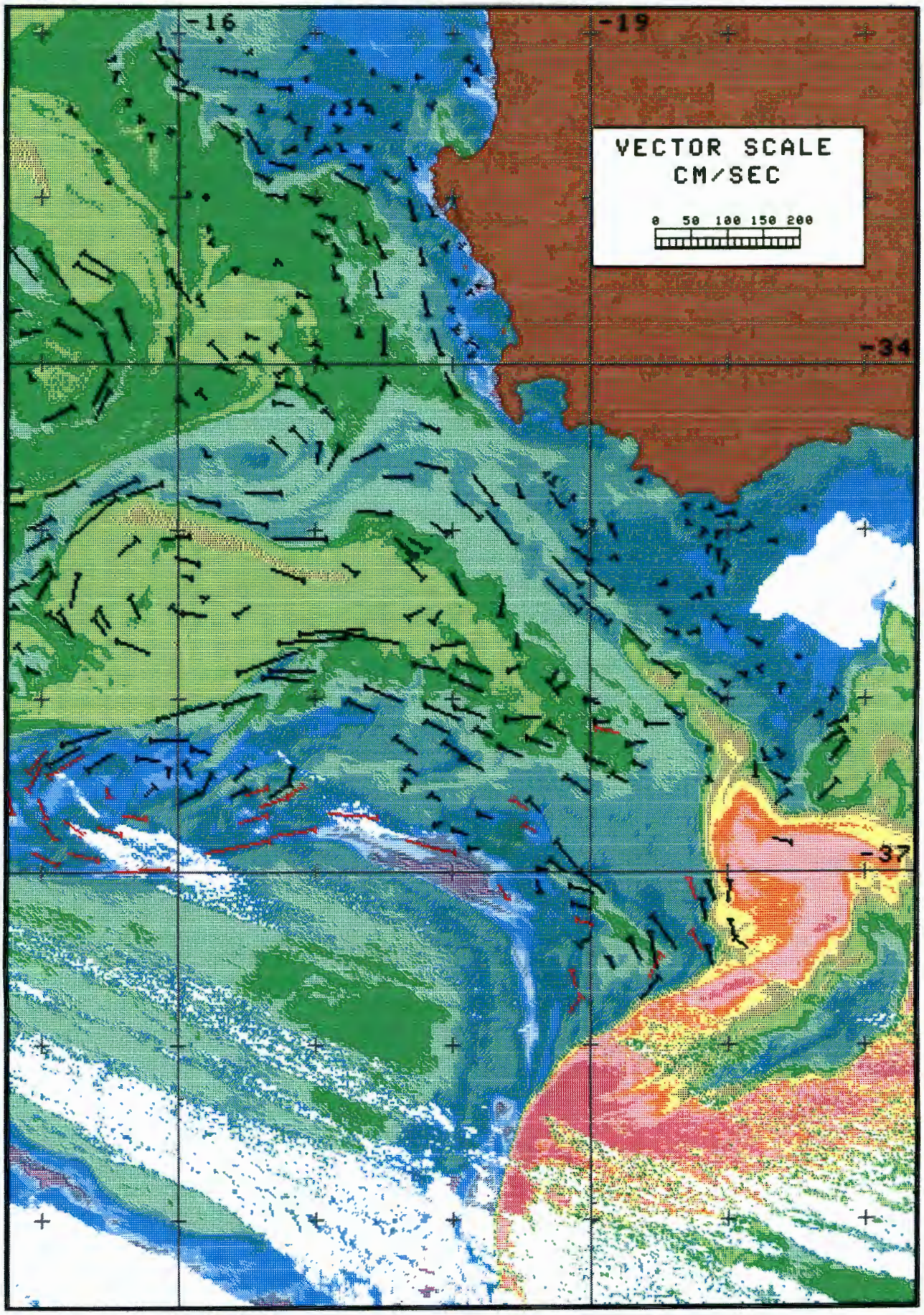


Fig. 4.8 The surface advection pattern in the eastern half of the study area. The black arrows were obtained from images IM2 and IM3 and the red arrows from images IM1 and IM3. The image is AVHRR band 4 of NOAA-11 on 14 May 1990.

paragraph. As far as possible, each feature shall be described under a separate heading but interaction between adjacent features makes it difficult to adhere rigorously to such a scheme and in some cases a degree of overlap and repetition is unavoidable.

The development of RING1

The most important observation made when comparing the two images (18 March and 14 May), is that at the time of the March image, the Agulhas Retroflection was positioned near 16° 14'E. This position corresponds approximately with the secondary retroflection point given as 16°E by Lutjeharms and van Ballegooyen (1988). The subsequent eastward retraction to 19°E almost certainly implies that a ring was shed from the retroflection (Section 4.1.2). This ring is probably seen in Figure 4.5 as the large elliptical shape centred approximately on 38°S : 16° 20'E. The feature has north/south and east/west diameters of about 230 and 350 km respectively, which compare well with the 307 km average diameter of Agulhas Current rings (Lutjeharms and van Ballegooyen *op. cit.*) and also with previously observed ellipticity in freshly formed rings (Lutjeharms and Gordon 1987, Olson and Evans 1986). The northern edge of the feature is seen to be near 37°S and that of the Retroflection loop near 38° 20'S (Fig. 4.5). The average northward drift component for the elliptical ring observed by Olson and Evans was 4,1 cm/s. At this rate the northward

displacement of the Retroflection loop, through 148 km, could have occurred in about 42 days. It therefore seems reasonable to accept that the feature - to be referred to as 'RING1' - is indeed an Agulhas ring which separated from the Current at the beginning of April 1990.

The development of filament STCF2

Having arrived at the conclusion that a ring was shed, the origin of filament STCF2 follows logically. Harris, Legeckis and van Foreest (1978), Lutjeharms (1981) and Lutjeharms and van Ballegooyen (1988) have all shown the formation of a cold wedge or filament during the ring shedding process. The anticyclonic entrainment of the filament around the periphery of a ring - as seems to have occurred on this occasion - has however not been reported yet, although it could be regarded as being analogous to the entrainment of warm filaments, such as reported by Lutjeharms and Valentine (1988). Velocities derived in STCF2, along the northern edge of RING1, range between 22 cm/s in the northeast, increasing to 96 cm/s at top centre and decreasing again to about 32 cm/s in the cyclonically curving, northwestern part. Olson and Evans (1986) found maximum velocities at a distance between 110 and 130 km from the centre of each of two rings observed during November 1983; the maximum velocity recorded in the most freshly formed of the two was nearly 90 cm/s. Although a suggestion of a warm annulus is seen in the interior of RING1 (Fig. 4.5), the interior contains very little

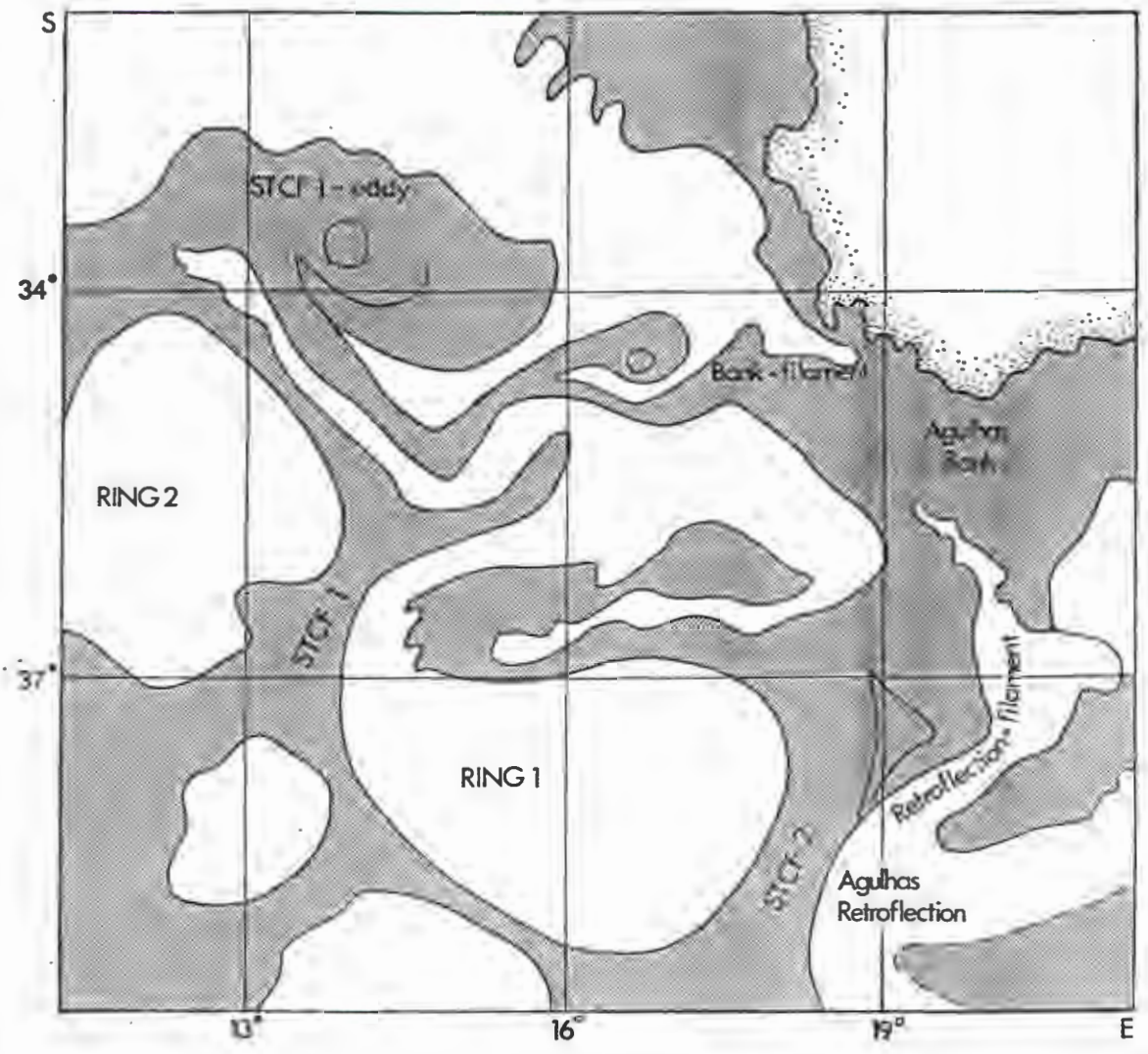


Fig. 4.9 A map of the positions of the main features discussed in the text.

thermal structure, which, in combination with cloudiness, made it impossible to obtain any advection data other than along the northern edge.

The second important observation made when comparing the 18 March and 14 May images, is that in both cases images a warm filament was attached to the northwestern tip of the Agulhas Retroflection

loop. In Figure 4.6 the filament is seen to bend northeastward and then curve northwestward along the western edge of the Agulhas Bank. In the image of 18 March the retroflection was located some two degrees of longitude further west and the attached filament much taller, but otherwise the structure of the retroflection bears a striking resemblance to the one in Figure 4.6. In fact, a very strong impression is conveyed that the

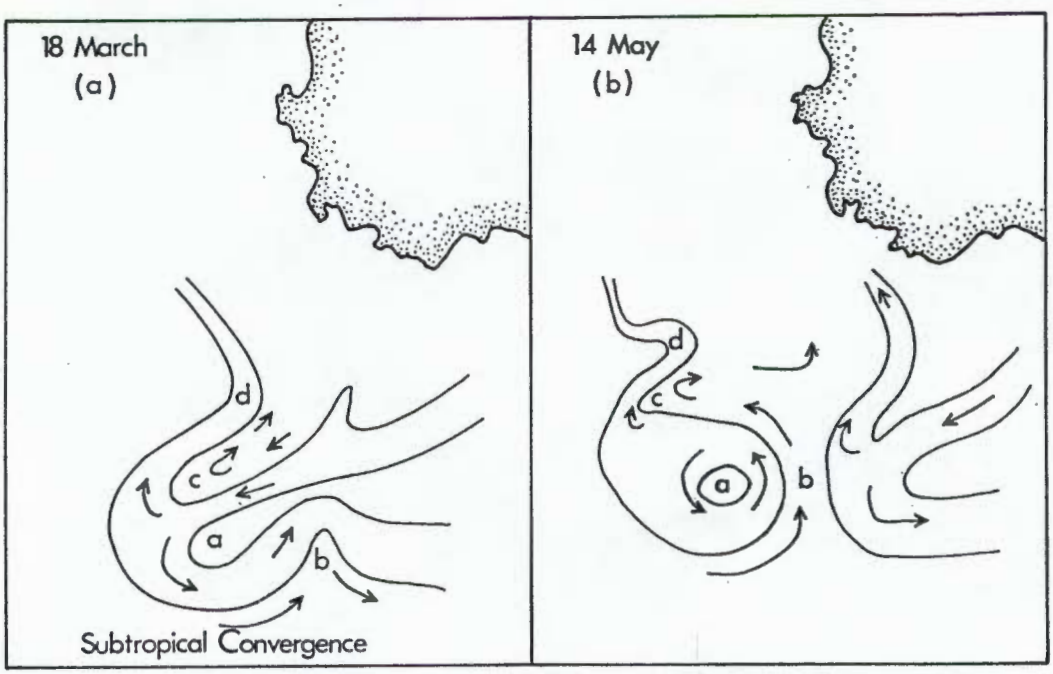


Fig. 4.10 A diagrammatic presentation of the Agulhas Current retroflection on 18 March (a) and 14 May 1990 (b). The sequence illustrates the separation of a ring (RING1) from the retroflection loop.

entire ensemble of features in the southeastern quadrant of Figure 4.6, consisting of :(a) RING1,(b) the cold cyclonically rotating feature between 36 and 37°S and (c) the warm anticyclone with its northwestward-stretching filament, was derived through truncation and west-northwestward transportation of a portion of the 18 March Agulhas Retroflection loop. This concept is

illustrated diagrammatically by Figure 4.10 and may be visualised by performing an imaginary shift of the retroflection loop in Figure 4.6. If the loop is shifted about one degree of latitude north and five degrees of longitude westward, a good match is obtained between several of the more obvious features. For example, the pool of cold water situated at about 38°S : 20-21°E would correspond with the western tip of the loop in STCF2 near 36° 30'S : 15-16°E ((c) in Fig 4.10) and the curving part of the Retroflection-filament, near 37°S : 20-21°E would correspond with the position of the body of warm water centred at about 35° 30'S : 16°E ((d) in Fig 4.10).

This correspondence has several interesting repercussions, one of which is that filament STCF2 would not be wholly composed of Subtropical Convergence Water entrained around the rim of RING1, as initially assumed. Instead, part of the structure, ie. the cyclonically rotating wedge between 36 and 37°S : 14° 30' - 17°E ((c) in Fig 4.10), was apparently derived from the cold water originally present in the elbow between the Agulhas Current loop and the northeastward curving filament. It is reasonable to assume that the westward flowing Agulhas Current and northeastward flowing filament would, in combination, have imparted cyclonic vorticity to this pool of water. Only when this water subsequently became linked with the Subtropical Convergence, during separation of RING1, would the observed flow have been established (Fig 4.10). In Figure 4.6 it is fairly certain that the pool of cool water between the Current and the retroflection filament (seen near 38°S : 20-21°E) is of

southeastern Agulhas Bank origin. However, it can not be said that the water which became incorporated into STCF2 was of the same source, since at that time the retroflection was located at least two degrees of longitude west from its position in Figure 4.6 and it is not known at what stage the filament developed. If the filament developed at a time subsequent to the Retroflection passing the southern tip of the Bank, the water would most likely have been South Atlantic surface water. Lutjeharms and Stockton (1987) mapped the trajectories and origins of such warm filaments and established a preferred origin near the southern tip of the Bank, but filaments originated along the entire retroflection range. If the filaments are related to plumes and meanders occurring on the Agulhas Current (Lutjeharms, Catzel and Valentine 1989), one may expect the filaments to develop as the retroflection passes the southern tip of the Bank and then migrate westward with the propagation of the retroflection. In such a case, it is probable that some Agulhas Bank water will always be entrained as in Figure 4.6. It is however not certain that this is the only mechanism for generating filaments.

RING2

In the third instance, the March and May images are compared in an attempt to understand the processes which gave rise to the filament STCF1 and its cyclonic eddy. The March image shows an intense cold filament in the same location as STCF1 but only reaching to about 36° 30'S; it's northern tip was curving in an easterly direction and rested against a large, warm, disk-shaped

feature centred at about 35° 30'S : 13° 40'E. The latter feature was partly cloud obscured, but seems to have had a diameter of 300 to 360 km. At the same time, the tip of the warm filament from the Agulhas retroflection was residing against the eastern side of the warm disk, and curved in an anticyclonic sense along its rim. Since both filaments, in contact with the warm feature, were describing anticyclonic trajectories, one assumes the object was rotating in a similar sense, which, when seen in conjunction with its diameter, suggests that this was perhaps a predecessor of RING1 - ie an older Agulhas ring - and this feature shall be referred to as 'RING2'. In Figure 4.6 a warm, circular feature of similar dimensions is found centred near 35° 30'S : 12° 30'E. The position of this feature, relative to the one seen in the March image, is such that it would have meant a westward propagation of only 2 cm/s, much less than the 3,8 cm/s observed by Olson and Evans (1986) for their 'Cape Town ring', but compares very well with the theoretical translation rate of 2,1 cm/s, computed by the same authors. If RING2, in the March image, was to be translated at a velocity equivalent to that actually observed by Olson and Evans, no suitable counterpart is found in the image from 14 May. Therefore it is assumed that the two features are the same. Whether it is indeed a ring is not certain, but evidence points that way, ie. it has the correct dimensions, it is the warmest feature in Figure 4.6 other than the Agulhas Current itself, it is apparently rotating in an anticyclonic sense with velocities of up to 90 cm/s (Fig.4.7) (similar to the maximum velocity found by Olson and Evans) and it has survived for at least two months.

The development of filament STCF1

The situation portrayed by the image of 14 May is now, as far as filament STCF1 is concerned, almost identical to the one described diagrammatically by Lutjeharms and van Ballegooyen (1988) where a tall filament of Subtropical Convergence water developed between adjacent Agulhas rings. It also seems clear that the extreme length of STCF1 is a consequence of entrainment along the periphery of RING2. In the image of 18 March the tip of the embryonic STCF1 was , as already said, residing along the southern quadrant of RING2's rim and was curving eastward. At the same time the warm filament was positioned along the eastern rim quadrant and being swept anticyclonically northwards. Roughly two months later, ie. as seen in Figure 4.6, the western portion of the Retroflection loop had separated into a new ring, but the filament attached to it had remained mostly intact and was, together with the tip of STCF1, being wound around the rim of RING2 as can be seen near 35°S : 13-14°E.

The STCF1-eddy

The Bank filament, ie. the cold filament streaming westward from the northwesterly edge of the Agulhas Bank, and the STCF1-eddy both developed in totality within the two month period from 18 March to 14 May and no information is available about their development histories.The STCF1-eddy is slightly elliptical in shape with a long axis of about 270 km, in the

northwest/southeast direction and a short axis of 210 km; it is located approximately 400 km from the Cape Peninsula. It actually seems to contain both a cold core, near 33° 40'S : 14°E, and a warm core, near 34°S : 15°E, but the circulation is cyclonic - typical of a cold core eddy (Fig. 4.7). The cold core contains a small area of 16-16,5°C water which is about 1°C colder than any of the water being fed into it (Fig. 4.6). The feature represents an incredibly complex mixing centre where at least four different source waters are being mixed together (Fig 4.7) : (a) South Atlantic Subtropical Surface Water enters along the northeastern quadrant, (b) Agulhas Bank Water enters via the Bank filament, which in itself seems to contain contributions from the northwestern Agulhas Bank, the Agulhas Current and water from STCF2 consisting of Subtropical Convergence Water and perhaps some water from the eastern Agulhas Bank, (c) relatively pure Subtropical Convergence Water enter via filament STCF1 along the southwestern quadrant and (d) some modified Agulhas Current Water, originally present in the Retroflection filament prior to separation of RING1 ((d) in Fig 4.10). In the northeast, some water derived from the Cape Columbine upwelling plume is also being fed into the eddy, but Figure 4.8 shows a jet sweeping northwards around the Cape Peninsula near the thermal front, in a manner similar to that described in Case Study 1, so that it is not clear whether the water from Columbine is in fact upwelled water or transported Agulhas Bank water.

Velocities in the STCF1-eddy are typically 50-60 cm/s with a maximum of 70-80 cm/s along the southeastern quadrant, ie.

nearest to the rim of RING2. The ring itself also exhibits maximum velocities (70-90 cm/s) at this point, perhaps suggesting local acceleration through convergence. The average angular velocity, computed from vectors associated with the cold core rotation is 51 degrees/day and for the warm core, 56 deg/day but varies generally between 29 and 140 deg/day, which means that the difference between the cold and warm core averages is not significant. Vectors near the centres of the cold and warm cores are both 12 cm/s, pointing more or less northwards, which could indicate the drift of the feature as a whole.

The Bank-filament

The STCF1-eddy is centred on water of about 4400m depth, in a corner between the northwestwardly directed bathymetric contours of the continental rise and a deep sea floor ridge protruding zonally westward to about 11°E, between 35 and 36°S. The Bank filament feeding into the eddy, runs parallel to the northern edge of the ridge. The filament originates near 36°S : 20°E and runs northwestward as a well defined current which has its core directly above the shelf edge in water of about 300m depth (Fig. 4.8). Core velocities follow the shelf edge and increase from about 25-30 cm/s, near the inception point, to about 55-60 cm/s, near the tip of the warm water (35° 30'S), and the latter velocity is maintained until the filament starts to curve away from the continent (18°E). At this point some water diverge, with one branch flowing northwards as a gradually narrowing current

on the seaward side of the coastal thermal front. Velocities in this flow are fairly constant at 35-40 cm/s. Off Cape Columbine the flow diverges from the coast, following the edge of the cold water plume situated there, but does not decrease markedly in speed until it reaches the very northern end of Figure 4.8 where some water turns south again, presumably entrained in a flux of South Atlantic Subtropical Surface Water feeding into the STCF1-eddy. Two small cyclonic eddies are seen in the Columbine plume, probably induced by the shear effect of the frontal jet. Along the coastal section, Cape Peninsula to Columbine, two sets of vectors were obtained (Fig. 4.8). These run more or less parallel to each other with the inshore group on average only slightly weaker than the other. The impression is given of a coherent current, but may in fact represent two parallel flows, such as found in Case Study 1 (Section 4.2). The outer line of vectors coincides with the shelf edge and probably represent the shelf-edge jet (Section 4.1.1), while the second line of vectors are about 40 km nearer the coast - over the shelf and clearly associated with the thermal front, as was found in Case Study 1 as well. In Case Study 1 very much higher velocities - up to 88 cm/s - were found in the shelf-edge jet, which was on that occasion transporting warm water. The maximum velocity to be found in the jet, in Figure 4.8, is only 41 cm/s. Another conspicuous difference between the flows as depicted by Figures 4.4 and 4.8 respectively, is found in the nearshore pattern to the north of the Cape Peninsula. In the discussion of Figure 4.4, attention was drawn to the absence of the expected near shore poleward flow. In Figure 4.8 there is still no evidence of a

significant poleward flow, but the coherent equatorward drift seen in Figure 4.4 is now absent.

The composition of the Bank-filament is of interest from the point of view of possible removal of anchovy eggs and larvae from the Agulhas Bank spawning ground (Sections 4.1.1 and 4.2). Judging by the thermal structure (Figs 4.6 and 4.8), the filament seems initially to have transported mainly water derived from the STCF2 filament, but that it is also entraining warm water from the Retroflection-filament - which now covers the cold core south of about 35° 30'S - and also some water from the western Agulhas Bank and possibly even from the southeastern tip of the Bank. Water derived from the extreme northwestern part of the Bank seems to be diverted northwards along the seaward side of the West Coast thermal front and thus would aid the transportation of larvae to the West Coast recruitment grounds. However, it would appear that water entrained from other parts of the Bank, was mostly being advected offshore. This suggests that, were a situation such as portrayed by this case study to occur near the peak of the spawning period, it could have serious consequences for anchovy recruitment.

The westward flowing branch of the Bank-filament is more than 80 km wide at the point where it diverges from the coast, south of the Cape Peninsula, and has a core velocity of about 50 cm/s, decreasing to about 30 cm/s at the edges. As the filament progresses westward it narrows to less than 25 km at the 15°E meridian. Velocities increase in a westerly direction and reach

a maximum of 81-104 cm/s at the point where the filament is compressed between the STCF1-eddy (to the north) and the warm anticyclone (to the south). The velocity increase is not uniform, but crude estimates of the volume transport suggest that the velocity increase would be sufficient to compensate for the narrowing of the filament. The small cold core eddy seen in the filament, near 34° 30'S : 16° 30'E, has an average angular velocity of 54 deg/day - very similar to that obtained for the larger STCF1-eddy.

The Retroflection-filament

The northwestward flow along the shelf edge of the western Agulhas Bank may be a permanent or semi-permanent feature. Bang and Andrews (1974) suggested the existence of such a current as a mechanism ('conduit') for transporting Agulhas Current water to the region adjacent to the Cape Peninsula upwelling plume, but the matter has received little attention in the literature. Lutjeharms, Bang and Valentine (1981) made brief reference to a flow of about 52 cm/s (1 knot) which, according to their diagrams, originates near the southern tip of the Bank, as a branch of the Agulhas Current. The trajectories of Agulhas Current filaments along the western Bank tend to confirm the existence of such a current (Lutjeharms and Stockton 1987), and presents a scenario comparable with the entrainment of the Retroflection-filament seen in Figure 4.8. Measurements made during 12 cruises using a ship-mounted ADCP (Acoustic Doppler

Current Profiler) demonstrated a shelf-edge flow, as far south as 36°S, with velocities ranging between 25 and 75 cm/s at a depth of 30 meters (Boyd *et al.* in press). In Case Study 1 (Section 4.2) a northwesterly flow of 55-65 cm/s, transporting warm water, was detected at the edge of the Bank, from about 35°S northwards (Fig.4.4); these velocities agree well with the 54-56 cm/s drift rates of a drogue tracked by Shelton and Hutchings (1982) in that area. This second case study produced similar velocities in the area north of 35°S, but the current is seen to be transporting mainly cold surface water and can be traced further south to at least 36°S. Together the evidence seems to confirm Bang and Andrew's (1974) suggestion of a shelf-edge current along the western Agulhas Bank extending into the jet off the Cape Peninsula. Available data points towards at least a semi-permanent feature with velocities increasing to a maximum of about 50-75 cm/s along the northwestern Bank.

The advection structure of filament STCF2

Filament STCF2 originates in the Subtropical Convergence near 18° 10'E (Fig.4.8) - a position corresponding closely with the average longitude (18° 25'E) reported for such features (Lutjeharms and van Ballegooyen 1988). Its position between the Agulhas Retroflection and the freshly formed Agulhas ring (RING1) furthermore corresponds with Lutjeharms and van Ballegooyen's scenario for the separation of rings - more or less as portrayed by Figure 4.10. The curiously folded shape of the filament, as

was previously discussed, seems to be a consequence of a warm filament having existed on the Retroflection at the stage of separation of RING1. Velocities in this folded portion of STCF2 are in general very high. Those along latitude line 37°S range between 73 and 96 cm/s and conform with the velocities expected for the rim of a fresh ring (Olson and Evans 1986). Unexpectedly though, the most energetic motion is seen in the easterly directed flow along the northern edge of the filament near 36°S, between 16 and 18°E. In this region, where the flow is not directly driven by the anticyclonic rotation of RING1, four vectors larger than 90 cm/s were obtained, two of which were in excess of 100 cm/s. The cold water wedge ((c) in Fig 4.10) has a north/south diameter of 90-140 km and must therefore give rise to frictional forces equivalent to, or larger than those which would be encountered in the Agulhas Retroflection loop. Thus one could expect the formation of small (90-140 km diameter), highly energetic cold core eddies - ie. something analogous to the formation of Agulhas rings. The closed cyclonic circulation indicated by the vectors near 36° 30'S : 15° 30'E (Fig 4.8), suggests that the formation of such an eddy was well advanced at that point. Cyclonic closure could also be expected near 18° 30'E. The southerly to southeasterly flow near 37°S : 19° 30'E was one of the advection features that required checking through derivation of a second set of vectors. But, as shown by the similarity of the black and red arrows, there can be little doubt about the authenticity of the poleward flow. This could be either a consequence of the convergence of the fast, eastward directed flow in STCF2 with the southern Agulhas Bank bathymetry, or, more

probably, a compensation mechanism for the northward fluxes in STCF2 on the western side and the Retroflection-filament/Agulhas Bank shelf-edge jet on the eastern side.

The advection structure of filament STCF1 and the geometry of RING1

The tall filament of Subtropical Convergence water, STCF1, originates in a wedge-shaped structure positioned between 13 and 14°E, directly west of the southernmost of the two Agulhas Current rings, RING1. The other ring, RING2, is positioned to the west-northwest of the first, with STCF1 in between. Imbedded in the wedge, and centred on 38°S : 13°E, is a small warm core feature, some 70 km in diameter, which seems to impart some local anticyclonic vorticity. Lutjeharms and Valentine (1985) observed the incorporation of such features into the Subtropical Convergence and suggested anticyclonic rotation. This feature was detectable on the earlier 18 March image, which also showed another, larger (*ca.* 300 km), warm disk to the southeast - the northern rim of which is seen near 39°S : 14-15°E in Figure 4.8. Only three vectors could be obtained on the northwestern edge of this latter feature (Fig. 4.7, 39°S : 13° 30'E). These vectors range between 30 and 51 cm/s in a cyclonic, rather than the expected anticyclonic, sense. Also, on the eastern side of the wedge-shaped base of STCF1, the anticyclonic flow expected from contact with RING1 was not obtained. With the exception of the flow round the embedded warm eddy the advection in STCF1 is

directed northwards and northwestwards, in other words, consistent with a flow along the rim of the northerly of the two rings (RING2). The anticyclonic rotation of the more southerly ring (RING1) on the eastern side of the filament, however, seems to have only a minor influence on the advection pattern in the filament. On the western edge of the filament, where it is in contact with RING2, velocities range from 40-45 cm/s in the wedge, decreasing to a minimum of 33 cm/s in the narrow part near 35°S and then increasing to a maximum of about 90 cm/s near the top (34° 30'S : 12° 30'E). On the RING1 side of the filament, velocities are substantially less : 25-34 cm/s in the wedge decreasing to 11-20 cm/s in the narrow part near 36°S. This velocity gradient, however, is much less than would be expected - in fact a complete reversal in flow direction may have been anticipated. While lacking information on the velocity structure in RING1, we deduce from this that the part of RING1 in contact with the eastern edge of the filament is not exhibiting the expected anticyclonic velocity of *ca.* 90 cm/s seen along the northern rim of the ring (Fig. 4.8, 37°S : 16-18°E). This observation is apparently consistent with the asymmetric shape of the ring - in the sense that a zone of relatively low anticyclonic velocity should be associated with the warm filament leading off the northwestern quadrant of the ring. The freshly formed ring observed by Olson and Evans (1986) had roughly the same geometry as RING1, ie. it was elliptical with warm water being advected from the northern side of the ring. Trajectories of satellite-tracked drifters placed in the ring were reported diagrammatically by Olson and Evans, and may be interpreted to

support the idea of a low velocity zone on the western, elongated quadrant of the ring.

Reproducibility of the vector computation procedure

To conclude this study, a few words about data quality. Although the second set of vectors were computed primarily as a means of obtaining advection data in those parts of the image, which as a consequence of cloud cover could not be studied with the main vector set, some degree of spatial overlap between the two sets was achieved. Scrutinizing Figures 4.7 and 4.8, one can identify 12 locations where a direct comparison of the two sets was feasible - at least as far as the vector magnitudes are concerned. For these the mean difference in magnitude (main set minus auxiliary set) and RMS of the differences are: -1,3 cm/s and 6,4 cm/s respectively. These results are slightly better than, but similar to, the results obtained from Case Study 1.

4.4 CASE STUDY 3 : AN AGULHAS RING IN THE SOUTH EAST ATLANTIC AND ITS EFFECT ON THE SOUTHERN BENGUELA REGION. JUNE 1989.

Data and data processing :

Two images were used for feature tracking purposes :

IM1 : 15 June 1989 13:35 GMT . NOAA 11 Ascending pass.

IM2 : 16 June 1989 13:25 GMT . NOAA 11 Ascending pass.

Image processing was done in a manner identical to the procedure described for Case Study 2 (Section 4), which includes, separation of the study area into three sub-images for geometric transformation. The overall area of interest is shown in decimated form by Figure 4.11. Both images were almost entirely cloud free, other than in the south. Cloud cover thus set the southern limit for the study; the northern limit was set by the limit of the available imagery. Feature tracking was performed using MTRACK as in case studies one and two. A total of 745 vectors were derived and are illustrated by the two slightly overlapping images, Figure 4.12 (western section) and Figure 4.13 (eastern section). Image IM1 was used for the creation of all three figures. As in the other case studies use was made of the Institute's archive of photo-negative AVHRR imagery to trace the development of some of the major features observed in the case study images.

Discussion.

The genesis and known properties of Agulhas rings were sketched in Section 4.1.2 and also extensively discussed in Case Study 2 (Section 4.3). The particular ring to be studied in this section, is seen in Figure 4.11, centred approximately on 28° 40'S : 8° 50'E - 770 km west of the Orange River mouth (Refer to Fig. 4.1). The ring was first detected in April 1989 during a research cruise transect between Cape Town (34°S : 18°E) and Vema Seamount (31° 40'S : 8° 20'E) and was subsequently studied in more detail during a cruise in May 1989. A detailed account of the hydrographical aspects, based on the data from these cruises was given by Duncombe Rae *et al.*(1992a). These data, in combination with GEOSAT altimetry data, confirmed the Agulhas Current origin of the feature. Hydrographic data also showed that while the ring was well defined in sub-surface temperature profiles,it had little surface expression. During the April cruise,the core surface temperature was 21°C and during May, 19°C. The NOAA 11 temperature during June was 18-18,5°C (Fig 4.11) and the ring is detectable only by virtue of the filament of cold water wrapped around its perimeter. The hydrographic data also served to identify the cold water as mature upwelled water from the Benguela Current upwelling front.

Aspects of upwelling and upwelling filaments in the southern Benguela Current region were briefly discussed in Section 4.1.1. Lutjeharms *et al.*(1991) suggested two mechanisms for the

formation of exceptionally long filaments : one being interaction with Agulhas rings - as in this case study - and the other being 'berg winds'. These winds comprise a seaward flow of air, heated adiabatically during the passage from the continental plateau to sea level. Both images used in this case study showed signs of

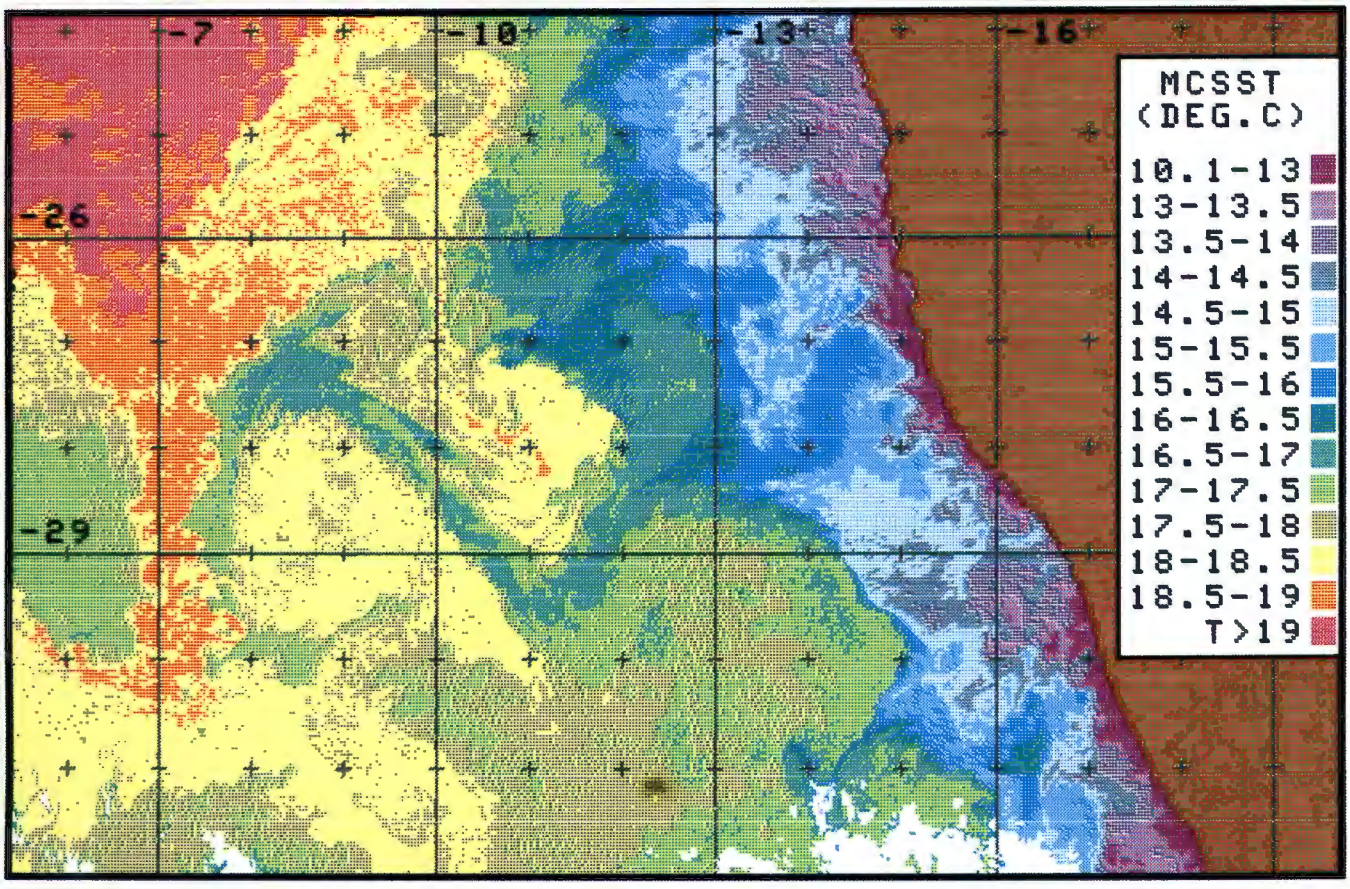


Fig. 4.11 The sea surface temperature distribution pattern in the case study area from a NOAA-11 AVHRR image of 15 June 1989.

berg wind activity in the form of dust plumes extending variable distances offshore.

The ring diameter in Figure 4.12, estimated on the outside of the cold filament, is about 330 km in the north/south and 250 km in the east/west direction. Measured on the inside of the filament, dimensions are about 230 and 190 km respectively. The estimated width of the filament along the ring perimeter is therefore 30-50 km. These estimates of ring diameter differ significantly from those described by Duncombe Rae *et al.*(1992a) - both as regards shape and orientation. Using the 16°C-isotherm at 200m during the May 1989 cruise, they arrived at a 165 km north/south and 330 km east/west dimension.

Using archive AVHRR imagery, the development of the thermal structure seen in Figure 4.11 can be followed back only as far as the 6th of June, but a warm core feature of similar dimensions, possibly representing a more southerly position of the ring can also be seen on an image of 30 April 1989. Comparison of the 6 June image with the 15 June image (from which Figure 4.11 was derived), suggests that the ring had indeed at that stage been more elongated in the east/west-axis than shown by Figure 4.11. Estimates of ring advection were obtained by overlaying the images and measuring the displacement of five recognisable points on the ring and filament during the nine days separating the June images:

(a) Southern tip of the ring near 30°S : 8° 30'E : NNE 5,5 cm/s

(b) Northern tip of the ring near 27°S : 8° 30'E : NNE 10,4 cm/s

(c) On the northeastern quadrant
of the ring near 28°S : 9° 30'E : NNE 6,2 cm/s

(d) The bend in the filament
at 29° 30'S : 11° 20'E : SE 7,8 cm/s

(e) Southern extremity of the
bend near 29° 40'S : 11° 20'E : SE 7,6 cm/s

From the hydrographic data Duncombe Rae *et al.(op.cit.)* inferred a translation rate of 6,4 cm/s in a northwesterly direction over the 42 day interval between the April and May cruises, and from the GEOSAT data spanning a period of 34 days (March to April) a speed of 6,8 cm/s. The average translation rate, from the first three points above, is 7,4 cm/s - a fair agreement with Duncombe Rae *et al.*'s results. More significantly though, the estimates show a distinct difference in displacement rate for the northern and southern parts of the ring, which if assumed constant over the 24 days between the May cruise and the date of Figure 4.11, would have increased the north/south dimension by about 100 km. This is not enough to remove the observed dimensional differences but at least suggests that the differences are real, time-related, geometrical variations, and not artifacts of measurement or procedural inaccuracy.

The image of 30 April also shows a cold filament streaming linearly offshore in a west-southwesterly direction and circling

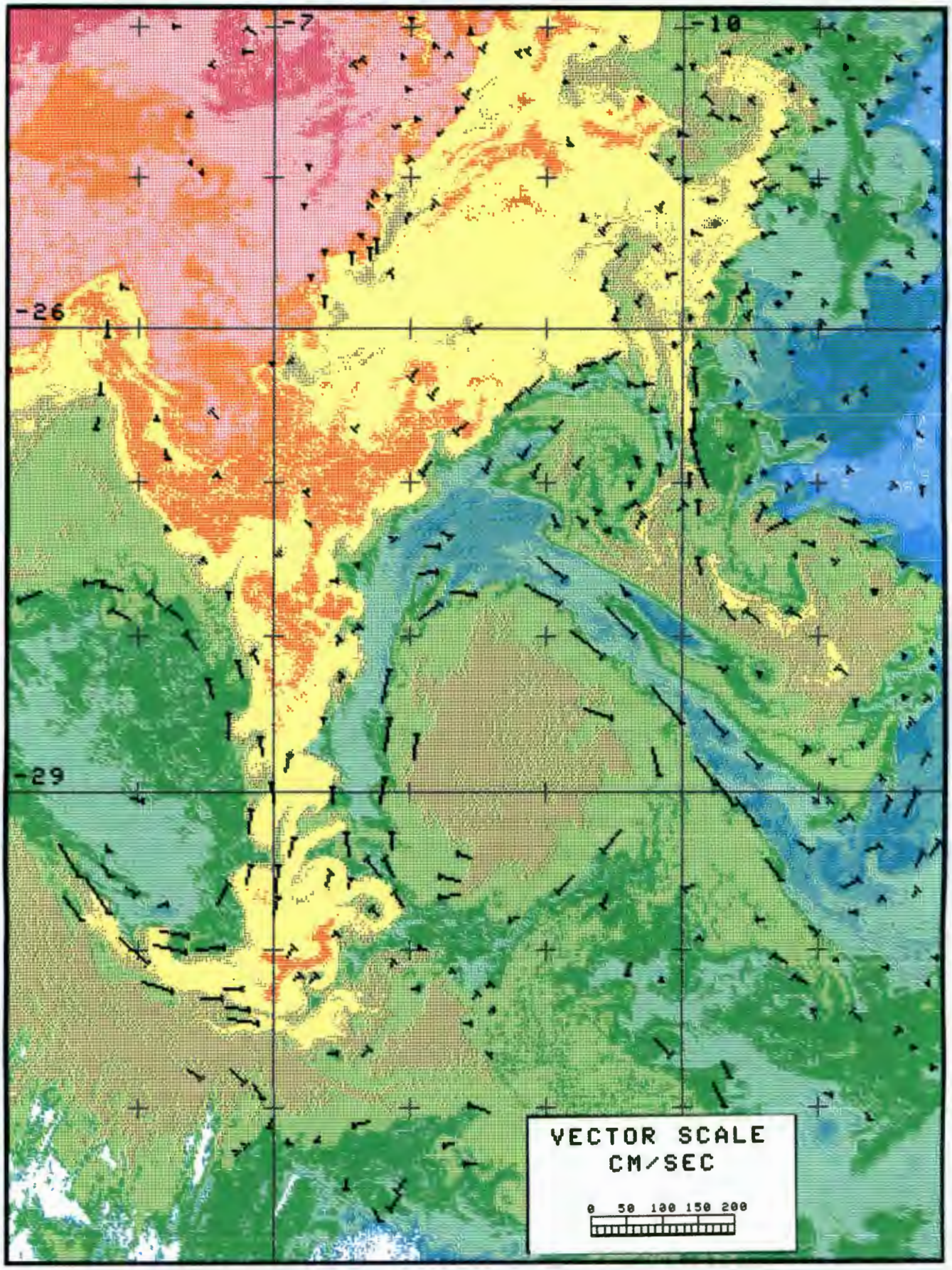
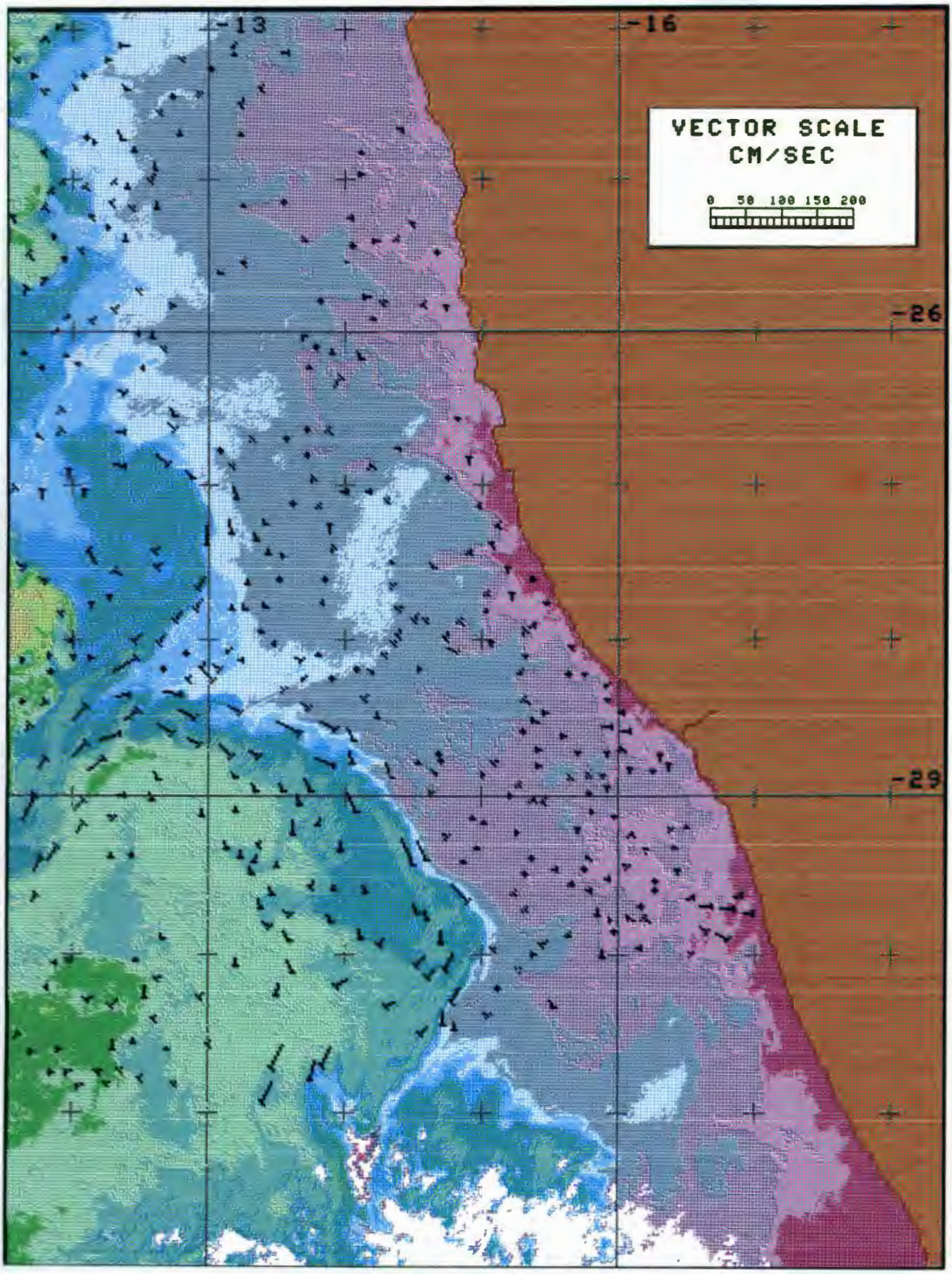


Fig. 4.12 The surface advection pattern over the western half of the study area. The image is AVHRR band 4 of the NOAA-11, 15 June 1989.



4.13 The surface advection pattern over the eastern half of the study area. The image is AVHRR band 4 of the NOAA-11, 15 June 1989.

anticyclonically round a somewhat elliptical warm feature centred at about 31° 9'S : 11° 8'E. Its dimensions were estimated as 280 km and 370 km in the north/south and east/west directions respectively, both measured on the outside of the filament. The thermal structure in the image is generally dissimilar to that in Figure 4.11, but the presence of the filament and the rough agreement of dimensions lead one to suspect it might have been the same ring. Furthermore, the latitudinal position corresponds more or less with the 24 April 1989 position of 31° 24'S obtained from GEOSAT altimetry data (Duncombe Rae *et al., op. cit.*). The translation rate estimated for the centre of the warm core, over the 46 days between 30 April and 15 June, is 8,1 cm/s in a northwesterly direction, and for a point on the peripheral filament near the northern tip of the ring 8,5 cm/s. Both estimates are somewhat faster than any of the previously mentioned estimates but this is thought to reflect the inaccuracy involved in defining specific positions on the two dissimilar thermal patterns.

In Figure 4.11, the cold filament is seen to originate on the thermal front near 28°S : 13°E and extend southwestward for a distance of 330 km, from where it bends northwestward for about 440 km to the northern tip of the ring. The SST structure creates the impression that the filament is advecting coastal, upwelled, water ie. water from the landward side of the thermal front. However, the vector pattern (Fig. 4.13) shows quite distinctly that, while some upwelled water is being entrained near the filaments base, most of the water being transported by the

filament is derived from the front itself or from directly seaward of the front; the latter probably being not pure oceanic water but, judging by the temperature (Fig. 4.11), mature upwelled water as reported by Duncombe Rae *et al.*(*op.cit.*). Virtually all water is entrained from the intense frontal region to the south of the filament. Velocities in the entrainment region seem to be largest in locations directly offshore of the most intense thermal gradient; typical values are 25-35 cm/s with a maximum of 46 cm/s. On the landward side, the fairly fast and orderly flow seen along the front changes rapidly to weak (0-10 cm/s) almost randomly-orientated currents. Four estimates of the rate of change across the front produced : 1,46 ; 1,4 ; 1,4 and 1,35 cm/s per kilometre, with a corresponding SST gradient of about 0,3-0,5 °C/km.

Comparison of the image of 15 June with AVHRR images received earlier in the month can help to clarify aspects of the thermal structure seen in the filament's base and entrainment region. The image of 6 June shows the filament to be attached to the plume of cold water (14,5-15,5 °C) which extends southwestward from the coast to about 28°S : 13° 30'E in Figure 4.11. At that time a second plume of cold water was present to the south, but in the period between the images the two plumes coalesced. The warmer water which originally separated the plumes can be seen in Figure 4.11 as the band of 15,5-16°C water stretching southwestward from the coast to the filament's base. From this observation one must deduce that while the vector pattern in Figure 4.13 shows very little or no entrainment of upwelled water from north of the

filament's base, it must have been drawing water mainly from that source at an earlier date. In fact, considering the small amount of upwelled water apparently being entrained at the time of this case study (Fig. 4.13), one is inclined to think most of the cold water in the filament was derived from the north rather than from the south.

From a comparison of the two images it is also possible to tell that the pool of relatively warm water (16,5-17,5 °C) with its centre near 27° 30'S : 12° 30'E in Figure 4.11, was derived, in a similar manner, through coalescence of the main filament with a smaller one - the remnants of which are still seen to the north. This pool of water seems to be spinning up into an anticyclonic warm core eddy with a diameter of about 80 km.

Velocities in the initial southwestward flowing part of the filament are similar to those in the entrainment region, ie. typically 25-35 cm/s (Fig 4.13), but increase from the point where the filament turns northwestward, so that velocities as high as 65-75 cm/s are found along the northeasterly quadrant of the ring, between 9 and 10°E (Fig. 4.12). The velocity field along this part of the filament is distinctly asymmetrical, with fast advection rates along the southwestern edge which decrease gradually towards the northeastern side. The decrease is evidently a frictional effect caused by the southward flow of warm water on the northeastern side of the filament. At the northern tip of the ring, the flow in the filament seems to be obstructed or blocked by the presence of the warm water (18-19°C)

present there. The filament is perceptibly wider at this point, some water is diverted into a small cyclonic eddy (*ca.* 115 km diameter) and the northwesterly advection drops to a maximum of 40 cm/s in this location. The flow then accelerates to a maximum of about 55 cm/s in the centre of the southward flowing portion of the ring, decelerates to about 30-35 cm/s in the vicinity of the divergence in the southwestern corner and finally accelerates to about 45 cm/s along the northeastward flowing quadrant. The highest geostrophic velocities computed by Duncombe Rae *et al.*(*op.cit.*) were southwestward 55 cm/s and northeastward 46 cm/s, which are similar to the velocities obtained with feature tracking along the western and eastern sides of the ring. Judging by the temperature section published by Duncombe Rae *et al.*, shipboard sampling was not extended far enough north to have encountered the filament on their northeast/southwest transect. Perhaps, for this reason they failed to detect the fast flow (up to 75 cm/s) shown by feature tracking to be associated with the filament along the northeastern quadrant of the ring - although one can not say whether such velocities were in fact present at the time of the cruise.

To the northeast of the ring, warm water is advected into a 'pocket' bounded by the main filament in the south and a shorter broad protrusion of cool water to the north. Water can be seen entering the pocket in a southward direction, through a narrow neck (12 km wide) at 26° 40'S : 10°E (Fig. 4.12), with typical velocities of 30-40 cm/s. Most of the water then flows southeastward for about 200 km, presenting in effect a return

flow, parallel to the offshore directed flow in the filament, but finally ending up rotating in an anticyclonic sense. The velocity in the return flow is maximally 30 cm/s and the flow gives rise to small cyclonic eddies of about 10-20 km diameter. Two of these are located along the interface of the two flows, at 28° 10'S : 10° 30'E and near 28° 20'S : 11° 40'E.

The situation portrayed here resembles the one described by Flament *et al.*(1985) for a Californian upwelling filament. The Californian filament was 50 km in width with typical offshore velocities of 55 cm/s; the velocity in the return flow, on the equator side of the filament, was 35 cm/s and the shear-edge eddies 10-15 km in diameter. However, in spite of these similarities it would be unwise to consider the two situations analogous. The 'return flow' displayed by Figure 4.12, in particular, seems to be of a transient nature and not representative of a true counter flow, as implied by the Californian study. According to Figure 4.11, temperatures in the core of the anticyclone, fed by the return current, are as high as 18,5-19°C. Water of such high temperatures is available only in quantity some 300 km to the northwest, with no indication that any of it is being advected southwards (Fig.4.12). In fact, water entering the anticyclone seems to be derived from no further north than about 150 km (ie. just south of the 26°S latitude). Furthermore, as indicated by the small plume of cold water extending southwards from the eastern side of the neck (at 27° 30'S : 10° 30'E) the water entering the pocket consists at least partially of entrained upwelled water. The above observations

suggest that the onshore flow may have been more active prior to the date of the case study image but is now in the process of being pinched off. This suggestion is supported to an extent by a fairly cloudy image received on 17 June (two days after the Figure 4.11 image); the image shows the edge of the cold eddy at the northern tip of the ring had moved approximately 10 km eastward. This should have effectively shut off the flow into the anticyclone, but due to cloud patches in that area, one is unable to tell with certainty. Never the less it seems fairly certain that closure was in the process of taking place and thus would have given rise to a warm eddy imbedded in the upwelling front. Such eddies are commonly observed all along the West Coast upwelling front (Lutjeharms and Stockton 1987), but in this instance, the formation of the eddy could probably be related, albeit indirectly, to the northward motion of the Agulhas ring, rather than generic frontal processes.

The most intriguing feature seen in this case study imagery, is undoubtedly the large, cold core, gyre to the west of the Agulhas ring (Fig. 4.12). The gyre is centred near 28° 45'S : 5° 50'E; it is elliptical in shape with a long axis of about 270 km in the north/south direction and a short axis of about 200 km in the east/west direction, ie. it is slightly smaller, but comparable in size to that of the Ring itself. Flow along the rim of the feature is cyclonic at a fairly uniform rate of 30-40 cm/s along the northern, eastern and southern sides but more than 60 cm/s along the southwestern edge. Warm water (18,5-19°C) is drawn southwards between the gyre and the Agulhas ring and wraps round

the rim of the gyre, creating a thermal 'negative' of the Agulhas ring with it's encircling cold filament. The surface of the gyre is nearly isothermal, 17-17,5°C (Fig. 4.11). Water of such temperature can be seen directly seaward of the upwelling front, located 700 km to the east of the gyre, and also about 200 km to the south near 32°S : 7-8°E and one is inclined to suspect that the water in the gyre was derived from this latter source rather than from the more distant frontal region. However, the image of 6 June provides some evidence to suggest that this suspicion may be false. Although fairly cloudy, the image shows the presence of the gyre with sufficient clarity that its dimensions may be estimated as 290 km (north/south) and 240 km (east/west). At that stage the cold filament only encircled the Agulhas ring as far as its southwestern 'corner', from where it extended westward towards the cold gyre. One can therefore say with fair certainty that the gyre consists, at least partially, of water derived from the filament. The scenario is structurally very reminiscent of the STCF1-filament/STCF1-eddy combination (Case Study 2, Section 4.3) where the eddy also formed adjacent to an Agulhas ring and was clearly composed of water from the Subtropical Convergence- and Agulhas Bank filaments, being rotated into a tight, elongated , spiral. It is not really possible to tell from the 6 June image whether the filament was also describing a spiral in this instance, but the presence of traces of warm water, observed inside the gyre, suggest such a possibility.

Duncombe Rae *et al.*(1992a) estimated that the Agulhas ring could have had an influence on the Benguela region for a period of one

to two months and in that period entrained an estimated 0,5 x 10^{13} m^3 of water in the annulus around the ring - a volume of water which is of the same magnitude as the Benguela frontal region to a depth of 100m. As seen in Figure 4.13 and described above, water entrained in the filament seems to be derived primarily from the frontal zone itself rather than the broad expanse of upwelled water on the landward side of the front. This is important from the perspective of the transportation of young anchovies and other species from the Agulhas Bank to the West Coast nursery grounds. In Case Study 1 (Section 4.2) we discussed the advection pattern along the West Coast and concluded that the frontal zone offered a favourable route for the northward transportation/migration of larval fish. Removal of a large volume of water from this zone could therefore, in principle, have had an adverse effect on the recruitment success of some commercially important fish species. In fact, Duncombe Rae *et al.*(1992b) suggested that a marked decline in anchovy abundance between November 1988 and November 1990 was partly attributable to the passage of the Agulhas ring during 1989. This argument seems to be further strengthened by the fact that in estimating the volume of water drawn offshore (Duncombe Rae *et al.* 1992a), the presence of the cold core gyre was not taken into consideration. The surface area of the gyre, as seen in Figure 4.12 is about 169700 km^2, which equates to a volume of 1,7 x 10^{13} m^3, if we follow Duncombe Rae *et al.* in using 100m as the depth estimate. Even if allowance is made for the probability that the feature was not wholly composed of Benguela Current frontal zone water, its presence should still significantly increase the

estimated volume of water removed from the frontal region and hence the potential adverse effect on the biota of the region.

5. SUMMARY

In this section the results obtained in chapters 2, 3 and 4 will be reviewed. In chapters 2 and 3 a number of tests were carried out to evaluate various aspects of the image registration and feature tracking procedures; the results from these tests are reviewed in 'conclusion' sections imbedded in those chapters. Only the most prominent of those results will be repeated here together with some comments regarding the performance of the procedures as experienced during the applications phase.

No review sections were included in the case studies in chapter 4 and therefore the results from those sections will be more comprehensively summarised and discussed here. At various points within the case study discussions, computed vector data could be compared with velocities previously reported in the literature; such comparisons will receive specific attention since no other means of assessing the accuracy of the satellite derived velocities is available.

5.1 Chapter 2. Image registration.

The work in this chapter was aimed at the establishment of a procedure for transforming AVHRR imagery to Mercator projections, as a preparatory step for feature tracking operations and consisted of two parts ie. (a) development of an image navigation routine for the computation of image coordinates (line and pixel numbers) for given geographical coordinates (latitudes and

longitudes) and *vice versa*, and (b) the development of a transformation procedure based on the navigation routine.

The navigation routine.

The navigation routine is contained within program NOANAV (Appendix A) and is based on a combination of elliptical orbit and ellipsoidal earth models. Orbit parameters are obtained from TBUS-bulletins and are assumed static during the orbital revolution which yielded the relevant image. One or more ground reference points are needed for the computation of the nodal longitude, λ_o, and nodal time (Note that nodal time was not used explicitly but rather the travel time, t_s, from the equator to the first line in the image). Program NOANAV was designed to function for both ascending and descending satellite passes and also on all parts of the globe.

By using a pair of test images, three aspects of the navigation routine was evaluated :

(a) Precision of the reference points and the effect there-of on λ_o and t_s.

The test results suggested the need for selecting several reference points, preferably within the pixel range 300 to 1750, and to discard outliers. Under these circumstances λ_o could be obtained with a standard deviation of 0,005° and t_s with a standard deviation of 0,08 sec. If expressed in terms of image coordinates, this means the standard deviations are equivalent to about half a pixel and half a scan line.

(b) Behaviour of the navigation algorithms in the satellite flight direction.

The question that was addressed by this test, was that if λ_0 and t_s were computed from a given reference point, how would navigation accuracy vary along the satellite's track. The matter was evaluated with five data sets (four from an ascending pass and one from a descending pass). Results were variable but suggested that at worst the position will deviate by one scan line and one pixel over a distance of about 500 lines.

(c) Behaviour of the navigation algorithms in the scan direction.

The question asked in this case, was the same as in (b) except that we were interested in the across-track direction rather than the along-track direction. Four data sets were used in the test (two each from an ascending and descending pass). The descending pass produced substantially larger errors than the ascending pass, but in general errors were of similar magnitude as in (b), ie. a maximum of one line deviation over a distance equivalent to 600 pixels and a maximum of one pixel over a distance of 400 pixels.

The transformation procedure.

To effect the transformation to Mercator projection, the navigation routine was imbedded in program GCPFIL (Appendix B) which is used to set up a data file of 'ground control points'

(GCPs) which is accessed by one of the existing ARIES II image processing software modules to compute a polynomial transform equation. The transform equation is later used in a cubic convolution resampling strategy to effect the transform to Mercator projection.

Empirical tests were carried out with 1st, 2nd and 3rd order polynomials computed from various numbers of GCPs (15 to 40). The best results were obtained with the 3rd order function. Precision was fairly independent of the number of GCPs used, provided 20 or more were used. The precision of the transform as a function of position within the image was evaluated in a subsequent test. The test indicated relatively small errors in the line number which remained approximately constant over the width of the image. Pixel errors however tended to increase rapidly when within about 400 pixels from the edge of the image. The latter effect could be reduced to acceptable levels by reducing the width of the sub-image (the area that was being transformed) to about 300 pixels. As a consequence of this result, the transformation of large case study images was handled in a piece-wise fashion, requiring the transformation of several sub-images.

In a final test, the overall transformation accuracy was evaluated; five test-images were used for this purpose. The transformation error, expressed as the root-mean-square of the differences between the actual and computed (transformed) positions of a number of ground reference points, were as follows:

Range of RMS line errors = 0,33 - 0,57

Range of RMS pixel errors = 0,65 - 0,92

Range of RMS latitude errors = 0,0048 - 0,0082 degrees

Range of RMS longitude errors = 0,0107 - 0,0154 degrees

From these we estimate that the maximum velocity component errors due to navigation factors that could be expected from a feature tracking operation which uses two images 12 hours apart (assuming a combination of a positive error in one image with a negative error in the other) would be :

for the north/south component = 4 to 5 cm/s

for the east/west component = 6 to 7 cm/s

5.2 Chapter 3. Feature tracking

In this chapter the first step was to implement the automatic feature tracking procedure described by Emery *et al.*(1986). The procedure makes use of two-dimensional Fourier transforms and storage requirements for the transform arrays proved difficult to meet on the available image processing system. As a consequence of the storage problem, program ADVECT (Appendix C) which was written to test this procedure, turned out to be impractically slow. As a second step, program AUTOTR (Appendix D) was produced. The procedure in this case differed from the previous only in the sense that the tracking was performed without the use of Fourier transforms.

With AUTOTR processing time for the computation of one vector (in a 64x64 search area), was reduced to 3 min 50 sec compared with 7 min 15 sec through ADVECT. This was seen as good enough to

warrant further evaluation of AUTOTR's performance. For evaluation purposes a set of 88 vectors were derived manually from a pair of NOAA-9 images, 11 hours apart, and compared visually with those obtained by application of the automated procedure. The results were disappointing; at least one out of every three vectors were judged to be incorrect. Circumstances leading to incorrect results were analysed and it seemed that the strong thermal gradient associated with the warm Agulhas Current present in the test image, combined with the deformation of features along the boundary of the fast current, were responsible for about half the failures. No practical solution could be found for the problem. As a consequence of the results obtained in the test, the advantages and disadvantages of the automated procedure were reviewed in light of experience gained during the experiment and it was decided that the disadvantages, coupled with the limitations of the available computer hardware, outweighed potential benefits from the procedure.

Since automation of vector computation could not be achieved satisfactorily, the final stage in the development of a feature tracking procedure, consisted of the implementation of an alternative 'semi-automated' procedure. This procedure is in essence a manual process from which was removed as much as possible of the usual labour. The principle of 'template matching' which forms the basis of vector computation in the automated procedure, was retained as a labour saving device in the new process (program MTRACK, Appendix E).

An experiment indicated a time saving of 20 to 40% on the manual procedure and in the worst case, the average time per vector computation with MTRACK was 1 min 5 sec, which is substantially faster than the 3 min 50 sec achieved with the best of the two automated procedures tested.

5.3 Chapter 4. Applications

In this chapter the semi-automated procedure MTRACK was applied to three case studies, as a test for the procedure and as a novel approach for the quantification of large scale surface flow around southern Africa. The main findings are summarised below.

Case study 1. 20 July 1989. Figures 4.2, 4.3, 4.4 and 4.5

i) The study covers the Benguela region between 27 and 36°S and features an intrusion of warm water from the south.

ii) The intrusion appears in the case study imagery as a convoluted mass of warm water between 34° 30'S and 36°S, southwest of the Cape Peninsula. The flow pattern in this region is complex and advection velocities of up to 140 cm/s are observed.

iii) A warm filament, 20-40 km wide, is attached to the body of intruding water and runs roughly parallel to the coast between Cape Agulhas and Cape Columbine.

iv) Thermohaline data obtained from a research cruise, tentatively identifies the warm water as of Agulhas Current origin. This is confirmed by the thermal imagery.

v) The core of the filament is generally positioned just

seaward of the shelf edge. The most probable depth of the feature was determined as 100-150 m in a position 60 km southwest of the Cape Peninsula. It is proposed that the warm water is transported by a shelf-edge jet current.

vi) To the south of Table Bay (34°S) the warm water flow is in the form of a broad band with a core velocity of about 45 cm/s in a northwesterly direction. At 34°S the flow encounters a cool filament of water, extending westward from the coast. At this point the flow divides into two parallel longshore streams.

vii) The seaward side stream flows more or less along the shelf edge as far as Cape Columbine and transports warm water from the Agulhas intrusion. Velocities in this stream are typically 40 cm/s with a maximum of 88 cm/s at the divergence point. At Cape Columbine the flow moves off the shelf edge and runs in a northwesterly direction along the outer limits of the cool coastal water.

viii) The landward of the two streams seem to be concentrated along the thermal front. In the Cape Peninsula to Cape Columbine region the flow is parallel to the coast and about 20 to 40 km offshore. Typical velocities in this region are 32-34 cm/sec but acceleration to 40-44 cm/s takes place off the two Capes. In the region north of Cape Columbine the flow field is modulated by tongues and plumes of cool upwelled water protruding from the coast. There is a clear tendency for advection to take place parallel to the isotherms in these features, giving rise to offshore transport along their poleward sides and onshore transport

on the equatorward sides. Advection velocities are typically 20-30 cm/s.

ix) Virtually all the cool water plumes seen in the case study images terminate in partially detached cyclonic eddies. Six or seven such eddies are discernable. These are spaced at fairly regular intervals; the mean spacing is 156 km with a standard deviation of 24 km. All eddies but for the one off Cape Columbine seemed to have remained stationary over a period of eight days.

Case study 2. 14 May 1990. Figures 4.6 to 4.10

i) This study covers a section of the South East Atlantic between 32 to 39°S and 11 to 21°E and the imagery features a complex sea surface temperature (SST) pattern.

ii) By comparing the May 1990 case study image with an image received about two months earlier on 18 March, processes and events which gave rise to the observed pattern could be traced.

iii) The comparison helped to identify one of the thermal features as a new Agulhas ring.

iv) The new ring is roughly elliptical in shape, which agrees with previous observations of a new ring. The dimensions of the feature, as well as the advection velocities along the rim (maximum of 96 cm/s) also agree with previous findings (Olson and Evans 1986).

v) It was concluded that much of the complexity in the observed SST pattern was brought about by the shedding of the ring, and in particular, by virtue of the existence of

a warm filament on the retroflection loop at the time of ring separation.

vi) The presence of the filament at the time of ring separation seems to have had a number of consequences :

(a) it probably explain the elliptical nature of the ring,

(b) it may provide a mechanism for the transportation of water from the southeastern Agulhas Bank into the Atlantic,

(c) it may possibly provide a mechanism for the formation of highly energetic cyclonic and anticyclonic eddies of diameter 100-200 km.

vii) Evidence was also found which led to the tentative identification of an older Agulhas ring. Velocities obtained along the edge of this structure indicate anticyclonic rotation at a rate of about 90 cm/s - similar to the maximum velocity previously reported for Agulhas rings (Olson and Evans *op.cit.*).

viii) It was concluded that a long filament of cold Subtropical Convergence water, observed in the case study imagery, was a product of entrainment along the periphery of the second ring.

ix) The cold filament terminated in a large cold core cyclonic eddy with diameter 210-270 km. The eddy was located about 400 km west of the Cape Peninsula. No information could be obtained on the development history of the feature. Velocities in the feature were typically 50-60 cm/s (maximum 70-80 cm/s) and the average angular rotation rate was estimated at 51 degrees/day.

x) Judging by the thermal and advection structures, the eddy acted as an important mixing centre where at least four different source waters are mixed together.

xi) The eddy was seen to draw water from the northwestern edge of the Agulhas Bank from where the water flows westward in the form of a narrowing filament. Velocities in this filament are high, typically 50 cm/s but reaching a maximum of 104 cm/s.

xii) The advection pattern found in this case study supports the results of case study 1 in respect of a northwesterly shelf edge jet along the western Agulhas Bank. In case study 2 the flow can be observed to entrain warm Agulhas Current water. The flow rate seems to increase from about 25 cm/s near the southern tip of the Bank to a maximum of 50-60 cm/s along the northwestern edge of the Bank.

xiii) The advection pattern in the coastal region between the Cape Peninsula and Cape Columbine suggests two parallel, longshore, equatorward currents, one along the shelf edge and the other along the thermal front. This is in agreement with the conditions observed in case study 1 except that velocities in case study 2 are smaller. Velocities in the two branches are similar, about 30-40 cm/s.

xiv) Off Cape Columbine the shelf edge current diverges sharply from the shelf to flow in a northwesterly direction along the thermal front, in association with a plume of cool upwelled water located there.

Case study 3. 15 June 1989. Figures 4.11, 4.12 and 4.13

i) This case study covers a section of the South East Atlantic adjacent to the west coasts of northern South Africa and southern Namibia, between latitudes 24-32°S and longitudes 6-18°E. The imagery features an Agulhas ring centred approximately 770 km west of the Orange River mouth.

ii) The Agulhas Current origin of the feature was previously confirmed by a combination of hydrographic- and GEOSAT altimeter data (Duncombe Rae *et al.*1992a).

iii) SSTs in the ring are similar to the ambient South East Atlantic surface water and the ring is detectable in the imagery only by virtue of a filament of cool water wrapped around the ring's perimeter. Hydrographic data previously obtained served to identify the water in the filament as mature upwelled water (Duncombe Rae *et al.*1992a).

iv) The width of the filament (along the perimeter of the ring) varies between 30 and 50 km.

v) Ring translation rates were estimated by comparing the case study imagery with a pair of images received on earlier dates. The comparisons yielded translation rates of 7,4 and 8,3 cm/s which are somewhat higher than the 6,8 cm/s that was estimated from hydrographic data.

vi) The SST structure in the case study imagery creates the impression that the filament is advecting coastal upwelled water, ie. water on the landward side of the thermal front. However, the vector pattern shows quite distinctly that most of the water is derived from the front itself or directly seaward of the front. This result is confirmed by

salinities of 35,2 x 10^{-3} - typical of frontal water - measured along the northwestern edge of the ring (Duncombe Rae *et al.* 1992a)

vii) Velocities in the filament were typically 40-50 cm/s in an anticyclonic sense, but reached a maximum of 75 cm/s along the northeastern quadrant of the ring.

viii) A cold core eddy of similar dimensions as the Agulhas ring, was observed directly west of the ring. Warm surface water was drawn southwards between the ring and the eddy and is wrapped in a cyclonic sense along the rim of the eddy. Flow along the rim of the eddy was fairly uniform in speed at 30-40 cm/s but reaches a maximum of 60 cm/s along the southwestern quadrant.

ix) An image received prior to the case study imagery, suggests that the cold eddy consists at least partially, of water derived via the same filament from the frontal region.

5.4 Precision and accuracy of the velocity computation procedure.

Precision.

In case study 1 an experiment was conducted to assess the repeatability or precision of the vector computation procedure. For this purpose two independent vector sets were computed from the same pair of images. The vector sets were compared visually and 63 'pairs' selected on the basis that they seemed to be close enough in space to be directly comparable. Velocities in this subset ranged from 6 to 140 cm/s with an average of 32 cm/s. For

each pair, the second vector was subtracted from the first in terms of magnitude (speed) and direction. This gave average size and directional differences of 1,3 cm/s and 5° respectively. The root-mean-square (RMS) differences were larger, ie. 8,2 cm/s and 31°. This suggests a fairly even distribution of positive and negative values (differences) and hence the absence of bias. The high directional RMS is attributable to the poor directional resolution at small vector sizes, so that elimination of the six smallest vectors from the calculation, results in a reduction of the directional RMS to 12,4°. The RMS for vector magnitude is close to the error which would arise from a one-pixel error in definition of the vector coordinates.

In case study 2 advantage was taken of a situation where cloud cover had previously necessitated the use of two different image pairs to complete the description of the advection pattern, and the vectors were found to overlap in some areas. A set of 12 vector pairs as close as possible in space to each other, were selected from within the overlapping area. Nevertheless because of the tight rotation in the current field only the magnitudes of the vectors were compared. The average difference between the two sets is -1,3 cm/s and the RMS difference, 6,4 cm/s.

The above result is very similar to the one from case study 1 and leads us to conclude that the precision in vector magnitude is about 6-8 cm/s, and the directional precision - for all but the smallest vectors (*ca.*6 cm/s) - about 12°.

Accuracy.

Systematic deviations from the true velocity could arise from inaccuracies in the image registration procedure. This aspect was investigated fairly extensively in chapter 2 but results were rather variable. It was clear however, that errors could be expected to increase with distance from the ground reference points used in the navigation procedure. The pixel or longitude error seemed to be more pronounced than the line number or latitude error, particularly so when the edges of the image were being approached. This much was confirmed by the final test, in which the overall transformation accuracy was assessed. On the basis of this test, it was concluded that for a time separation of 12 hours between images, the maximum expected velocity component error would be about 6-7 cm/s and the maximum expected east/west error about 4-5 cm/s.

No evidence of systematic errors was found by visual inspection of the case study imagery. Such errors could however be difficult to detect, except perhaps as anomalous cross-isotherm flow patterns, but nothing of the kind was seen. In low velocity regions ($v<10$ cm/s) vectors tended to be very random in orientation, but this is most likely a precision problem rather than anything else.

On a few occasions during the discussion of the case studies, comparisons between the computed velocities and published results were possible, and since these comparisons represent the only means presently available for assessing the accuracy of the

derived results, we summarise them below.

i) The shelf-edge jet off the Cape Peninsula. Bang and Andrews (1974) recorded an equatorward surface flow of 60-80 cm/s. In case study 1, velocities of 63-88 cm/s were obtained in this position but in case study 2 a maximum velocity of only 41 cm/s was found. Bang and Andrew's results were obtained during the summer upwelling period when the thermal front - and hence the jet - could be expected to be stronger than the winter-time images used in both case studies. The presence of warm water in the first case study and the absence there-of in the second, could explain the different velocities.

ii) Flow along the northwestern edge of the Agulhas Bank. Shelton and Hutchings (1982) reported 54-56 cm/s drift rates for a drogue tracked in that area. Both case studies 1 and 2 indicated velocities between 55 and 65 cm/s.

iii) The divergence at Cape Columbine. Shannon (1985) reported velocities of 60 cm/s for both the branches of the divergence, but mentions that the nearshore branch was time variable and that both poleward and equatorward currents have been recorded. Of the case studies, only the first demarcated the divergence clearly and indicated equatorward velocities of 45 cm/s and 18-32 cm/s in the nearshore and offshore branches respectively. As in the case of the jet off the Cape Peninsula (i) one would expect the offshore branch to be stronger during the summer upwelling period.

iv) The mean flow rate in the Benguela Current north of Cape

Columbine. Shannon (1985) indicated 25 cm/s as typical for the region. In case study velocities in this region ranged between 20 and 30 cm/s.

v) Typical velocity ranges for all the above shelf regions were given by Boyd *et al.*(1992) based on measurements made during 12 cruises using a vessel-mounted Acoustic Doppler Current Profiler, and agreement with velocities derived in case studies 1 and 2 is good.

vi) Velocities along the rim of the newly formed Agulhas ring. Olson and Evans (1986) reported anticyclonic motion of nearly 90 cm/s. In case study 2, anticyclonic velocities of up to 96 cm/s were found along the edge of RING1.

vii) Velocities along the rim of the Agulhas ring discussed in case study 3. Duncombe Rae (1992a) reported the maximum southwestward geostrophic velocity as 55 cm/s and the maximum northeastward velocity as 46 cm/s. In case study 3 the maximum velocity computed in the southward flowing part of the filament (along the rim of the ring) was 55 cm/s and 45 cm/s in the northeastward flow.

These comparisons are seen as a semi-quantitative assessment of the procedure and agreement with published results is generally quite good and hence instills confidence in the validity of the procedure.

6.CONCLUDING REMARKS

The geometric correction procedure is fairly cumbersome but generally worked well, except for the fact that as it stands, only rectangular image sections are accepted as input. The output is also a rectangular frame but the actual data area appears as a roughly diamond-shaped section within the frame (eg. see Figures 3.1 and 3.2). Ordinarily one does not wish to present the image in that form, and thus would trim off substantial sections to get rid of the blank areas. This is a waste of time since a much larger area is transformed than actually used in the end and steps will be taken to streamline the process.

The concept of a 'semi-automated' procedure as programmed in MTRACK was found to work very satisfactorily. Cursor (pointwise) and window (template matching) modes were used interchangeably when computing vectors for the case studies. We found it easier to use the original AVHRR band 4 images with the cursor mode and TGM images with the window mode. Very little use was found for the window rotation facility.

From an oceanographic point of view, the single most prominent impression gained, was undoubtedly the influence of the Agulhas Current on the southern Benguela region. Agulhas rings have received a fair amount of attention in the literature during recent years but mostly in terms of the global heat balance. The

more direct effect of rings or other features derived from the Current, however, seems not to have been sufficiently recognised by marine environmentalists concerned with the Benguela ecosystem. There is a tendency among these scientists to view variation in the Benguela as being controlled entirely by events in the Atlantic and South East Atlantic in particular. This study suggests that events in the Indian Ocean, as transmitted by the Agulhas Current, may play a significant role in such highly important issues as the recruitment success of fish species in the Benguela region. One feels that this is a matter which should receive attention in future research.

REFERENCES

AGENBAG, J. J. and L. V. SHANNON 1988 - A suggested physical explanation for the existence of a biological boundary at 24°30'S in the Benguela system. *S. Afr. J. mar. Sci.* 6: 119-132.

AMMANN, C. J. 1984 - The Fourier Transform and related topics. In *Workshop on Image Processing (Part 1) July 23-24 1984*. Sartori-Angus, A. G. (Eds). *University of Natal, Durban, South Africa.*

BANG, N. D. 1971 - The southern Benguela Current region in February, 1966. 2. Bathythermography and air-sea interactions. *Deep-Sea Res.* 18(2): 209-224.

BANG, N. D. 1976 - On estimating the ocean mass flux budget of lateral and cross circulation of the southern Benguela upwelling system. *Internal Report, National Research Institute for Oceanology* SEA IR7616: 14 pp. + 4 Figures.

BANG, N. D. and W. R. H. ANDREWS 1974 - Direct current measurements of a shelf-edge frontal jet in the southern Benguela system. *J. mar. Res.* 32(3): 405-417.

BOUDRA, D. B. and E. P. CHASSIGNET 1988 - Dynamics of Agulhas

retroflection and ring formation in a numerical model. 1. The vorticity balance. *J. phys. Oceanogr.* **18**(2): 280-303.

BOUDRA, D. B. and W. P. M. DE RUIJTER 1986 - The wind-driven circulation of the South Atlantic-Indian Ocean. 2. Experiments using a multi-layer numerical model. *Deep-Sea Res.* 33(4): 447-482.

BOYD, A. J. 1983 - Intensive study of the currents, winds and hydrology at a coastal site off central South West Africa, June/July 1978. *Investl Rep. Sea Fish. Res. Inst. S. Afr.* 126: 47 pp.

BOYD, A. J., TAUNTON-CLARK, J. and J. P. J. OBERHOLSTER 1992 - Spatial features of the near-surface and midwater circulation patterns off western and southern South Africa and their role in the life history of various fish species. *S. Afr. J. mar. Sci.* 12(In press).

BRUNDRIT, G. B. 1981 - Upwelling filaments in the Southern Benguela Region. *Trans. R. Soc. S. Afr.* **44**(3): 309-313.

BRUSH, R. J. H. 1985 - A method for real-time navigation of AVHRR imagery. *IEEE Trans. Geosci. Remote Sens.* **GE-23**: 876-887.

CHAPMAN, P., DUNCOMBE RAE, C. M. and B. R. ALLANSON 1987 - Nutrients, chlorophyll and oxygen relationships in the surface layers at the Agulhas Retroflection. *Deep-Sea Res.*

34(8A): 1399-1416.

CHASSIGNET, E. P. and D. B. BOUDRA 1988 - Dynamics of Agulhas retroflection and ring formation in a numerical model. 2. Energetics and ring formation. *J. phys. Oceanogr.* **18**(2): 304-319.

CHERKIS, N. Z., FLEMING H. S. and J. M. BROZENA 1989 - Bathymetry of the South Atlantic Ocean 3°S to 40°S. *The geological Society of America Inc., Boulder, Colorado*: Map and Chart Series **MCH-069**.

CLOWES, A. J. 1954 - The temperature, salinity and inorganic phosphate content of the surface layer near St. Helena Bay,1950-52. *Investl Rep. Div. Fish. S.Afr.* **16**: 47pp.

CRAWFORD, R. J. M., SHANNON, L. V. and D. E. POLLOCK 1987 - The Benguela ecosystem. 4. The major fish and invertebrate resources. In *Oceanography and Marine Biology. An Annual Review* **25**. Barnes, M. (Ed.). Aberdeen; University Press: 353-505.

DARBYSHIRE, M. 1963 - Computed surface currents off the Cape of Good Hope. *Deep-Sea Res.* **10**: 623-632.

DARBYSHIRE, M. 1966 - The surface waters near the coasts of southern Africa. *Deep-Sea Res.* **13**: 57-81.

DEACON, G. E. R. 1937 - The hydrology of the Southern Ocean. *"Discovery" Rep.* **15**: 1-124 + Plates 1-44.

DE DECKER, A. H. B. 1970 - Notes on an oxygen-depleted subsurface current off the west coast of South Africa. *Investl Rep. Div. Sea Fish. S. Afr.* **84**: 24 pp.

DE RUIJTER, W. P. M. 1982 - Asymptotic analysis of the Agulhas and Brazilian Current systems. *J. phys. Oceanogr.* **12**: 361-373.

DUCK K. I. and J. C. KING 1983 - Orbital mechanics for remote sensing. *Manual of Remote Sensing, second ed., American Society of Photogrammetry*: 699-717.

DUNCAN, C. P. 1967 - Current measurements off the Cape Coast. *Fish. Bull. S. Afr.* **4**: 8-14.

DUNCAN, C. P. and J. H. NELL 1969 - Surface currents off the Cape coast. *Investl Rep. Div. Sea Fish. S. Afr.* **76**: 19 pp.

DUNCOMBE RAE, C. M., SHILLINGTON, F. A., AGENBAG, J. J., TAUNTON-CLARK, J. and M. L. GRÜNDLINGH 1992a - An Agulhas ring in the South Atlantic Ocean and its interaction with the Benguela Upwelling Frontal System. *Deep-Sea Res.* (In press).

DUNCOMBE RAE, C. M., BOYD, A. J. and R. J. M. CRAWFORD 1992b - 'Predation' of anchovy by an Agulhas ring : a possible

contributing cause of the poor yearclass of 1989. *S. Afr. J. mar. Sci.* 12 (In press).

EMERY, W. J., THOMAS, A. C. and M. J. COLLINS 1986 - An objective method for computing advective surface velocities from sequential infrared satellite images. *J. Geophys. Res.* 91(C11): 12865-12878.

ESCOBAL, P. R. 1976 - Methods of orbit determination. *Robert E. Krieger Publishing Company, Huntington, New York*: 314 pp.

GARCIA, C. A. E. and I. A. ROBINSON 1989 - Sea surface velocities in shallow seas extracted from sequential Coastal Zone Color Scanner satellite data. *J. Geophys. Res.* 94(C9): 12681-12691.

GILLOOLY, J. F. and N. D. WALKER 1984 - Spatial and temporal behaviour of sea-surface temperatures in the South Atlantic. *S. Afr. J. Sci.* 80(2): 97-100.

GONZALES, R. F. and P. WINTZ 1977 - Digital image processing. *Addison-Wesley Publishing Company, Advanced Book Program / World Science Division, Reading, Massachusetts*: 431 pp.

GORDON, A. L. 1985 - Indian-Atlantic transfer of thermocline water at the Agulhas retroflection. *Science, N.Y.* 227(4690): 1030-1033.

GORDON, A. L. and W. F. HAXBY 1990 - Agulhas eddies invade the South Atlantic: evidence from Geosat altimeter and shipboard Conductivity-Temperature-Depth survey. *J. geophys. Res.* 95(C3): 3117-3125.

GRÜNDLINGH, M. L. 1977 - Drift observations from Nimbus VI satellite-tracked buoys in the southwestern Indian Ocean. *Deep-Sea Research.* 24(10): 903-913.

GRÜNDLINGH, M. L. and J. R. E. LUTJEHARMS 1979 - Large-scale flow patterns of the Agulhas Current system. *S. Afr. J. Sci.* 75(6): 269-270.

HAMPTON, I. 1987 - Acoustic study on the abundance and distribution of anchovy spawners and recruits in South African waters. In *The Benguela and Comparable Ecosystems.* Payne, A. I. L., Gulland, J. A. and K. H. Brink (Eds). *S. Afr. J. mar. Sci.* 5: 901-917.

HARRIS, T. F. W., LEGECKIS, R. and D. VAN FORE[E]ST 1978 - Satellite infra-red images in the Agulhas Current system. *Deep-Sea Res.* 25(6): 543-548.

HARRIS, T. F. W. and L. V. Shannon 1979 - Satellite-tracked drifter in the Benguela Current system. *S. Afr. J. Sci.* 75(7): 316-317.

HARRIS, T. F. W. and C. C. STAVROPOULOS 1978 - Satellite-tracked

drifters between Africa and Antarctica. *Bull. Am. met. Soc.* 59(1): 51-59.

HARRIS, T. F. W. and D. VAN FOREEST 1978 - The Agulhas Current in March 1969. *Deep-Sea Res.* 25: 549-561.

HO, D. and A. ASEM 1986 - NOAA AVHRR image referencing. *Int. J. Remote Sens.* 7: 895-904.

HOLDEN, C. J. 1985 - Currents in St Helena Bay inferred from radio-tracked drifters. *In South African Ocean Colour and Upwelling Experiment.* Shannon, L. V. (Ed.). Cape Town; Sea Fisheries Research Institute: 97-109.

HOLDEN, C. J. 1986 - Spatial and temporal scales of coastal currents in the St Helena Bay - Cape Columbine region. *M.Sc. thesis, University of Cape Town*: 189 pp.

HOLDEN, C. J. 1987 - Observations of low-frequency currents and continental shelf waves along the west coast of South Africa. In *The Benguela and Comparable Ecosystems.* Payne, A. I. L., Gulland, J. A. and K. H. Brink (Eds). *S. Afr. J. mar. Sci.* 5: 197-208.

HUTCHINGS, L., HOLDEN, C. [J.] and B. [A.] MITCHELL-INNES 1984 - Hydrological and biological shipboard monitoring of upwelling off the Cape Peninsula. *S. Afr. J. Sci.* 80(2): 83-89.

KIDWELL, K. B. 1983 - NOAA polar orbiter data (TIROS-N, NOAA-6, NOAA-7, and NOAA-8) users guide. *National Oceanic and Atmospheric Administration, National Environmental Satellite, Data, and Information Service, National Climatic Data Centre, Satellite Data Services Division, Washington, D.C.*: 101 pp.

KING, D. P. F., ROBERTSON, A. A. and P. A. SHELTON 1978 - Laboratory observations on the early development of the anchovy *Engraulis capensis* from the Cape Peninsula. *Fish. Bull. S. Afr.* 10: 37-45.

LAMBERTH, R. and G. NELSON 1987 - Field and analytical drogue studies applicable to the St Helena Bay area off South Africa's west coast. In *The Benguela and Comparable Ecosystems*. Payne, A. I. L., Gulland, J. A. and K. H. Brink (Eds). *S. Afr. J. mar. Sci.* 5: 163-169.

LEESE, J. A., NOVAK, C. S. and B. B. CLARK 1971 - An automated technique for obtaining cloud motion from geosynchronous satellite data using cross correlation. *J. Appl. Meteorl.* 10: 118-132.

LEGECKIS, R. and J. PRITCHARD 1976 - Algorithm for correcting the VHRR imagery for geometric distortions due to the earth curvature, earth rotation, and spacecraft roll attitude errors. *NOAA Technical Memorandum NESS 77, Washington D.C.: 33 pp.*

LUTJEHARMS, J. R. E. 1981 - Features of the southern Agulhas Current circulation from satellite remote sensing. *S. Afr. J. Sci.* **77**(5): 231-236.

LUTJEHARMS, J. R. E. 1988 - Examples of extreme circulation events at the Agulhas retroflection. *S. Afr. J. Sci.* **84**: 584-586.

LUTJEHARMS, J. R. E., CATZEL, R. and H. R. VALENTINE 1989 - Eddies and other border phenomena of the Agulhas Current. *Continent. Shelf Res.* 9(7): 597-616.

LUTJEHARMS, J. R. E. and A.FOLDVIK 1986 - The thermal structure of the upper ocean layers between Africa and Antarctica during the period December 1979 to March 1979. *S. Afr. J. Antarct. Res.* 16(1)

LUTJEHARMS, J. R. E. and A. L. GORDON 1987 - Shedding of an Agulhas Ring observed at sea. *Nature, Lond.* 325(7000): 138-140.

LUTJEHARMS, J. R. E. and J. M. MEEUWIS 1987 - The extent and variability of South-East Atlantic upwelling. In *The Benguela and Comparable Ecosystems*. Payne, A. I. L., Gulland, J. A. and K. H. Brink (Eds). *S. Afr. J. mar. Sci.* 5: 51-62.

LUTJEHARMS, J. R. E., SHANNON, L. V. and L. J. BEEKMAN 1988 - On

the sea surface drift of the Southern Ocean. *J. mar. Res.* **46**: 267-279.

LUTJEHARMS, J. R. E., SHILLINGTON, F. A. and C. M. DUNCOMBE RAE 1991 - Observations of extreme upwelling filaments in the Southeast Atlantic Ocean. *Science, N.Y.* **253**: 774-776.

LUTJEHARMS, J. R. E. and P. L. STOCKTON 1987 - Kinematics of the upwelling front off southern Africa. In *The Benguela and Comparable Ecosystems.* Payne, A. I. L., Gulland, J. A. and K. H. Brink (Eds). *S. Afr. J. mar. Sci.* 5: 35-49.

LUTJEHARMS, J. R. E. and H. R. VALENTINE 1984 - Southern Ocean thermal fronts south of Africa. *Deep-Sea Res.* **31**(12A): 1461-1475.

LUTJEHARMS, J. R. E. AND H. R. VALENTINE 1988 - Evidence for persistent Agulhas rings south-west of Cape Town. *S. Afr. J. Sci.* **84**: 781-783

LUTJEHARMS, J. R. E. and H. R. VALENTINE 1988 - On mesoscale ocean eddies at the Agulhas Plateau. *S. Afr. J. Sci.* **84**(3): 194-200.

LUTJEHARMS, J. R. E. and R. C. VAN BALLEGOOYEN 1988 - The retroflection of the Agulhas Current. *J. phys. Oceanogr.* **18**(11): 1570-1583.

McCARTNEY, M. S. and M. E. WOODGATE-JONES 1991 - A deep-reaching anticyclonic eddy in the subtropical gyre of the eastern South Atlantic. *Deep-Sea Res.* 38(Suppl. 1): S411-S443.

NELSON, G. 1985 - Notes on the physical oceanography of the Cape Peninsula upwelling system. In *South African Ocean Colour and Upwelling Experiment.* Shannon, L. V. (Ed.). Cape Town; Sea Fisheries Research Institute: 63-95.

NELSON, G. and L. HUTCHINGS 1983 - The Benguela upwelling area. *Prog. Oceanogr.* 12(3): 333-356.

NELSON, G. and L. HUTCHINGS 1987 - Passive transportation of pelagic system components in the southern Benguela area. In *The Benguela and Comparable Ecosystems.* Payne, A. I. L., Gulland, J. A. and K. H. Brink (Eds). *S. Afr. J. mar. Sci.* 5: 223-234.

NINNIS, R. M., EMERY, W. J. and M. J. COLLINS 1986 - Automated extraction of pack ice motion from Advanced Very High Resolution Radiometer imagery. *J. Geophys. Res.* 91(C9): 10725-10734.

OLSON, D. B. and R. H. EVANS 1986 - Rings of the Agulhas Current. *Deep-Sea Res.* 33(1A): 27-42.

PEARCE, A. F. 1977 - Some features of the upper 500 m of the Agulhas Current. *J. mar. Res.* 35(4): 731-753.

ROSENFELD, A. and A. C. KAK 1982 - Digital picture processing. *Academic Press, New York.* 2 volumes : 435 pp + 349 pp.

SCHOWENGERDT, R. A. 1983 - Techniques for image processing and classification in remote sensing. *Academic Press Inc., New York* : 249 pp.

SHANNON, L. V. 1966 - Hydrology of the south and west coasts of South Africa. *Investl Rep. Div. Sea Fish. S. Afr.* 58: 22 pp. + 30 pp. of Figures.

SHANNON, L. V. 1985 - The Benguela ecosystem. 1. Evolution of the Benguela, physical features and processes. In *Oceanography and Marine Biology. An Annual Review* 23. Barnes, M. (Ed.). Aberdeen; University Press: 105-182.

SHANNON, L. V., HUTCHINGS, L., BAILEY, G. W. and P. A. SHELTON 1984 - Spatial and temporal distribution of chlorophyll in southern African waters as deduced from ship and satellite measurements and their implications for pelagic fisheries. *S. Afr. J. mar. Sci.* 2: 109-130.

SHANNON, L. V., LUTJEHARMS, J. R. E. and J. J. AGENBAG 1989 - Episodic input of Subantarctic water into the Benguela region. S. Afr. J. Sci. 85(5): 317-322.

SHANNON, L. V., LUTJEHARMS, J. R. E., WALKER, N. D. and J. J.

AGENBAG 1990 - A major perturbation in the Agulhas retroflection area in 1986. *Deep-Sea Res.* 37(3): 493-512.

SHANNON, L. V., NELSON, G. and M. R. JURY 1981 - Hydrological and meteorological aspects of upwelling in the southern Benguela Current. In *Coastal and Estuarine Sciences. 1. Coastal Upwelling.* Richards, F. A. (Ed.). Washington, D.C.; American Geophysical Union: 146-159.

SHANNON, L. V., STANDER, G. H. and J. A. CAMPBELL 1973 - Oceanic circulation deduced from plastic drift cards. *Investl Rep. Sea Fish. Brch S. Afr.* 108: 31 pp.

SNYDER, J. P. 1987 - Map projections - a working manual. *U.S. Geological Survey, Professional paper 1395.* U.S. Printing Office, Washington.

SVEDLOW, M., McGILLEM, C. D. and P. E. ANUTA 1978 - Image registration: similarity measure and processing method comparisons. *IEEE Trans. Aerosp. Electron. Syst.* AES-14(1): 141-150.

SVERDRUP, H. U., JOHNSON, M. W. and R. H. FLEMING 1942 - *The Oceans: their Physics, Chemistry, and General Biology.* Englewood Cliffs, New Jersey; Prentice-Hall: x + 1087 pp.

TOKMANIAN, R., STRUB, P. T. and J. McCLEAN-PADMAN 1990 - Evaluation of the maximum cross-correlation method of

estimating sea surface velocities from sequential satellite images. *J. atmos. oceanic Technol.* **7**: 852-865

VAN WOERT, M. 1982 - The subtropical front: satellite observations during FRONTS 80. *J. Geophys. Res.* **87**(C12): 9523-9536.

VISSER, G. A. 1969 - Analysis of Atlantic waters off the west coast of southern Africa. *Investl Rep. Div. Sea Fish. S. Afr.* **75**: 26 pp.

WAHL, D. D. and J. J. SIMPSON 1990 - Physical processes affecting the objective determination of near-surface velocity from satellite data. *J. Geophys. Res.* **95**(C8): 13511-13528.

WALKER, N. D. 1986 - Satellite observations of the Agulhas Current and episodic upwelling south of Africa. *Deep-Sea Res.* **33**(8A): 1083-1106.

WILSON, W. H., SMITH, R. C. and J. W. NOLTEN 1981 - The CZCS geolocation algorithms. *Scripps Institute for Oceanography, Ref. 80-13*: 37 pp.

APPENDIX A

Listing of program 'NOANAV'

```
NAME : NOANAV.FTN
A PROGRAM CREATED TO PERFORM GEOLOCATION ON NOAA AVHRR IMAGES.
THE PROGRAM WILL FUNCTION FOR BOTH ASCENDING (NORTH BOUND) AND
DESCENDING (SOUTH BOUND) SATELLITE TRACKS.  IT OPERATES IN ALL
HEMISPHERES. SOUTH LATITUDES AND EAST LONGITUDES ARE INDICATED
AS NEGATIVE.

NOANAV CALLS 6 SUBROUTINES :
  NOAN1    : CALCULATE THE EQUATOR CROSSING LONGITUDE AND TRAVEL
             TIME FROM EQUATOR TO CCT LINE NO. 1 FROM ONE OR MORE
             GROUND REFERENCE POINTS.
  NOAN2    : CALCULATES THE LAT AND LONG FOR A GIVEN LINE AND PIXEL NO.
  NOAN3    : CALCULATES THE CCT LINE AND PIXEL NO. FOR A GIVEN LAT
             AND LONG.
  NOAN4    : INPUT OF ORBITAL ELEMENTS AND SCANNER DATA.
  NOAN5    : CALCULATES CERTAIN CONSTANTS.
  NOAN6    : INTERPOLATES BETWEEN TWO SETS OF TBUS DATA FOR THE
             ORBITAL ELEMENTS RELEVANT TO A GIVEN IMAGE (TIME)

THREE DATA TABLES ARE USED :
1. XDATI(): WHERE ELEMENT
     1 = SEMI-MAJOR AXIS (KM)
     2 = NODAL PERIOD (MINUTES)
     3 = ORBIT INCLINATION ( DEGREES WESTWARD FROM EQUATOR. THIS IS
         A NEGATIVE QUANTITY FOR DESCENDING TRACKS)
     4 = ECCENTRICITY
     5 = ARGUMENT OF PERIGEE (DEG)
     6 = SCAN RATE ( SECONDS PER SCAN - IE CA. 1/6 OF A SEC)
     7 = DIGITIZATION STEP ANGLE -SCANNER ANGLE BETWEEN THE CENTRES
         OF SUCCESSIVE PIXELS (DEG)

2. XDATW() : THE SAME DATA AS IN XDATI() BUT IN RADIANS AND SECONDS.

3. XDATC() : WHERE
     1 = EQ. CROSSING LONGITUDE. FROM NOAN1 (RAD)
     2 = TRAVEL TIME FROM EQUATOR TO CCT LINE 1 - IN FLIGHT
         DIRECTION (SEC)
     3 = SATELLITE MEAN ANGULAR VELOCITY - 2*PI/NOD.PERIOD (RAD/SEC)
     4 = EARTH ROTATION RATE-ORBITAL PRECESSION RATE (FROM NOAN5 RAD/SEC
     5 = RELATIVE BEARING OF SCAN WITH RESPECT TO TRACK LINE ( SCAN
         SCEW EFFECT . FROM NOAN5. RAD)
     6 = TRAVEL TIME FROM PERIFOCUS TO EQUATOR ( NOAN5. SECONDS)
     7 = PI (RAD)
     8 = DEGREES TO RAD CONVERSION FACTOR (RAD/DEG)
     9 = THE INVERSE OF (3) IE NODAL PERIOD/2PI (SEC/RAD)

J.J.AGENBAG     S.F.R.I     JUNE 1989.  IN FORTRAN 77

PROGRAM NOANAV
IMPLICIT CHARACTER*1(C),REAL*8(X-Z)
CHARACTER AREA*3,DATAFN*10
DIMENSION XDATI(8),XDATW(8),XDATC(10)


INITIALISE DATA VARIABLES.

XDATI(1)=7229.6D0 ! RADIAL DISTANCE IN KM, TO SATELLITE
                  ! THIS IS THE SAME AS ORBIT SEMI-MAJOR AXIS.
```

```
      XDATI(2)=102.07D0 ! NODAL PERIOD IN MINUTES.
      XDATI(3)=99.04D0 ! ORBIT INCLINATION WESTWARD FROM EQUATOR.
      XDATI(4)=0.0D0           ! ECCENTRICITY OF ORBIT.
      XDATI(5)=0.0D0           ! ARGUMENT OF PERIGEE (DEG.)
      XDATI(6)=.16666667D0     ! SCAN RATE IN SECONDS PER SCAN LINE.
      XDATI(7)=.054431D0              ! DIGITIZATION STEP ANGLE (DEG)
C
      WRITE(5,20)
20    FORMAT(1X//' PROGRAM NOANAV : GEOLOCATION ROUTINE FOR NOAA',
     + ' SATELLITES :')
      WRITE(5,40)
40    FORMAT(1X/' ENTER 3 CHAR. AREA CODE FOR THE IMAGE :',$)
      READ(5,60)AREA
60    FORMAT(A3)
100   FORMAT(A1)
      DATAFN(1:3)=AREA
      DATAFN(4:10)='NAV.DAT'
C
C     PRESENT A MENUE OF OPTIONS.
C
200   CONTINUE
      WRITE(5,300)
300   FORMAT(1X,/1X,10X,'NOANAV MAIN MENUE :',/1X,10X,19('-'),/,
     + ' 1. ENTER ORBIT AND SCANNER DATA',/,
     + ' 2. OBTAIN ORBIT DATA BY INTERPOLATION OF TBUS INFORMATION',/,
     + ' 3. CALCULATE EQUATOR CROSSING LONGITUDE FROM REF. POINTS.',/,
     + ' 4. ENTER EQUATOR CROSSING LONG AND TRAVEL TIME DIRECTLY.',/,
     + ' 5. CALCULATE LAT. AND LONG. FOR A PIXEL.',/,
     + ' 6. CALCULATE LINE AND PIXEL NUMBERS FOR GIVEN LAT/LONG.',/,
     + ' 7. SAVE RELEVANT ORBITAL AND OTHER DATA IN A FILE.',/,
     + ' 8. RECOVER DATA PREVIOUSLY SAVED.',/,
     + ' 9. EXIT.')
340   WRITE(5,360)
360   FORMAT(1X/' ENTER MENUE ITEM :',$)
      READ(5,*)MENU
      IF(MENU.LT.1.OR.MENU.GT.9)THEN
          GOTO 340
        ELSE IF(MENU.EQ.1)THEN
          CALL NOAN4(XDATI)
          CALL NOAN5(XDATI,XDATW,XDATC)
        ELSE IF(MENU.EQ.2)THEN
          CALL NOAN6(XDATI)
          CALL NOAN5(XDATI,XDATW,XDATC)
        ELSE IF(MENU.EQ.3)THEN
          CALL NOAN1(AREA,XDATI,XDATW,XDATC)
        ELSE IF(MENU.EQ.4)THEN
          WRITE(5,400)
400       FORMAT(1X/' ENTER EQ.CROSSING LONG(DEG,-=EAST) AND TRAVEL'/,
     +     ' TIME(SEC) FROM EQ. TO LINE 1:',$)
          READ(5,*)XDATC(1),XDATC(2)
          XDATC(1)=XDATC(1)*XDATC(8)
        ELSE IF(MENU.EQ.5)THEN
          CALL NOAN2(XDATW,XDATC)
        ELSE IF(MENU.EQ.6)THEN
          CALL NOAN3(XDATW,XDATC)
        ELSE IF(MENU.EQ.7)THEN
          OPEN(1,FILE=DATAFN,STATUS='NEW',ERR=420)
          WRITE(1,*)XDATI,XDATW,XDATC
          CLOSE(1)
          GOTO 440
```

```
420           WRITE(5,430)DATAFN
430           FORMAT(1X/' UNABLE TO OPEN FILE ',A10,' : DATA NOT SAVED',/)
440           CONTINUE
            ELSE IF(MENU.EQ.8)THEN
              OPEN(1,FILE=DATAFN,STATUS='OLD',ERR=460)
              READ(1,*)XDATI,XDATW,XDATC
              CLOSE(1)
              GOTO 500
460           WRITE(5,480)DATAFN
480           FORMAT(1X/' FILE ',A10,' DO NOT EXIST.  DATA NOT RECOVERED',
     +        /)
500           CONTINUE
            ELSE
              GOTO 1000
          END IF
          GOTO 200
C
1000      CALL EXIT
          END
```

```
C
C       SUBROUTINE NOAN1.FTN
C       A SUBPROGRAM TO NOANAV. NOAN1 CALCULATES EQUATOR CROSSING LONGITUDE
C       AND TRAVEL TIME FROM THE EQUATOR TO SCAN LINE NO. 1
C
C       J.J.AGENBAG   S.F.R.I    JUNE 1989.   IN FORTRAN 77
C
        SUBROUTINE NOAN1(AREA,XDATI,XDATW,XDATC)
        IMPLICIT REAL*8(X-Z),CHARACTER*1(C)
        CHARACTER AREA*3,DATAFN*10,CDATE*9,CTIME*8
        DIMENSION XDATI(8),XDATW(8),XDATC(10),XSTOR(2,180),IDCSET(180)
C
        WRITE(5,100)
100     FORMAT(1X/' READ REF. POINT DATA FROM A FILE ? (Y/N):',$)
        READ(5,120)CINRP
120     FORMAT(A1)
        IF(CINRP.NE.'Y')GOTO 300
        WRITE(5,140)
140     FORMAT(' ENTER STORAGE FILE NAME(MAX=10 CHAR.) :',$)
        READ(5,160)DATAFN
160     FORMAT(A10)
        OPEN(1,FILE=DATAFN,STATUS='OLD',ERR=180)
        GOTO 220
180     WRITE(5,200)DATAFN
200     FORMAT(1X/' FILE ',A10,' NOT ACCESSABLE.')
        GOTO 1300        ! RETURN
220     WRITE(5,240)
240     FORMAT(' NUMBER OF REFERENCE POINTS ? :',$)
        READ(5,*)NUMRP
C
300     WRITE(5,320)
320     FORMAT(1X/' PRINT HEADER INFORMATION ? (Y/N) :',$)
        READ(5,120)CHAR
        IF(CHAR.NE.'Y')GOTO 400
        CALL DATE(CDATE)          ! GET DATE.
        CALL TIME(CTIME)          ! GET TIME
        WRITE(6,340)AREA,CDATE,CTIME,(XDATI(I),I=1,7)
340     FORMAT(1X/' NODAL LONGITUDE AND TRAVEL TIME FOR AREA ',A3,/,
     +  ' PROCESSED ON ',A9,2X,A8,'  WITH THE FOLLOWING PARAMETERS :',/,
     +  ' 1. SEMI-MAJOR AXIS (KM)',26X,'=',F12.5,/,
     +  ' 2. NODAL PERIOD (MINUTES)',24X,'=',F13.6,/,
     +  ' 3. INCLINATION ANGLE (DEG WESTWARD FROM EQUATOR) =',F15.8,/,
     +  '     ( A NEGATIVE INCLINATION INDICATES A DESCENDING TRACK)',/,
     +  ' 4. ECCENTRICITY OF ORBIT',25X,'=',F16.9,/,
     +  ' 5. ARGUMENT OF PERIGEE (DEG)',21X,'=',F14.7,/,
     +  ' 6. AVHRR SCAN RATE (SECONDS PER SCAN)',12X,'=',F15.8,/,
     +  ' 7. AVHRR DIGITIZATION STEP ANGLE (DEG)',11X,'=',F17.10,/)
400     WRITE(6,420)
420     FORMAT(1X/'  NO.    LINE',6X,'PIXEL',7X,'LAT.',6X,'LONG.',7X,
     +  'EQ.LONG',7X,'TIME',/)
C
C       SOME CONSTANTS.
C
        XRN=DSIGN(1.0D0,XDATI(3))          ! 1=ASCENDING, -1=DESCENDING TRACK.
        X2PI=2.0D0*XDATC(7)
        X90=0.5D0*XDATC(7)                 ! PI/2 RADIANS
        XDR=XDATC(8)
        XC1=(1.0D0-XDATW(4)**2.0)*XDATW(1)
        XINC=(XDATI(3)-XRN*90.0D0)*XDATC(8) ! INCLINATION W OF N (RAD)
```

```
      DO 440 I=1,40
         DO 430 J=1,2
            XSTOR(J,I)=0.0D0   ! ARRAY USED FOR MEAN AND ST. DEV CALC
430      CONTINUE
440   CONTINUE
C
C ---------------------------------------------------------------
C
C     START OF CALCULATION CYCLE.
C
C     INPUT REF. POINTS.  NB! SOUTHERN AND EASTERN COORDINATES ASSUMED
C     INPUT AS POSITIVE NUMBERS. PROGRAM CHANGE TO NEGATIVE.
C
      NUM=0              ! COUNTER FOR NUMBER OF SETS READ.
500   IF(CINRP.EQ.'Y')THEN     ! INPUT FROM A FILE.
         IF(NUM.EQ.NUMRP)GOTO 800
         READ(1,*)XLIN,XPIX,XLAT,XLON
        ELSE
         WRITE(5,520)NUM+1
520      FORMAT(1X/1X,I3,': ENTER LINE,PIXEL,LAT AND LONG(0=EXIT):',
     +    $)
         READ(5,*)XLIN,XPIX,XLAT,XLON
      END IF
      IF(XLIN.EQ.0.0.AND.XPIX.EQ.0.0)GOTO 800
      NUM=NUM+1
      XLAT=-XLAT
      XLON=-XLON
      XLATR=XLAT*XDR   ! REF. LATITUDE IN RADIANS
      XLONR=XLON*XDR   !  ,,  LONGITUDE ,,   ,,
      XLATR=DATAN(0.9933D0*DTAN(XLATR))  ! CONVERT GEODETIC LATITUDE TO
                                         ! GEOCENTRIC.
C
C     ITERATE TO FIND THE GROUND POINT LATITUDE. INITIALISE TO PIXEL LAT.
C     TERMINATE ITERATION WHEN DIFERENCE BETEEN TWO RESULTS ARE <=0.00004 RAD
C
      IRD=1              ! FIRST CALCULATION
      XNU=(XPIX-1024.5D0)*XDATW(7)     ! SCAN ANGLE TO PIXEL (RAD)
      XPSI=XDATC(5)    ! RELATIVE BEARING TO PIXEL =CA 90.01DEG IN RAD
      XRAD=6378.137D0*(1.0D0-.00335D0*(DSIN(XLATR))**2.0) ! EARTH RADIUS
                                                  ! AT REF PIXEL
      XLA=XLATR          ! INITIALISE GP. LAT TO REF. LAT
      XLAO=XLA           ! SAVE PREVIOUS LAT. FOR ITERATION CHECK.
540   XTETA=XRN*DASIN(DSIN(XLA)/DCOS(XINC))  ! ANGULAR DISTANCE EQ. TO XLA
      IF(XRN.GE.0.0D0)THEN               ! ASCENDING TRACK
         XV=X2PI+XTETA-XDATW(5) ! TRUE ANOMALY=ANG. PERIFOCUS TO XLA
         IF(XV.GE.X2PI)XV=XV-X2PI
        ELSE                             ! DESCENDING TRACK.
         XV=XDATC(7)+XTETA-XDATW(5)
      END IF
      IF(XV.LT.0.0D0)XV=XV+X2PI
      XCV=DCOS(XV)
      XINCO=DASIN(DSIN(XINC)/DCOS(XLA))        ! INCLINATION AT GP LAT.
      XPSIP=XPSI-XINCO          ! TRUE BEARING TO PIXEL
      XRD=XC1/(1.0D0+XDATW(4)*XCV)   ! GEOCENTRIC DIST. TO SATELLITE
                                     ! AT THIS LATITUDE.
      XALTC=XRD/XRAD             ! ALTITUDE FACTOR
      XTHS=XNU
      XTH=DASIN(XALTC*DSIN(XTHS))-XTHS   ! GEOCENTRIC ANGLE - GP TO PIXEL
C     WRITE(6,550)IRD,XLIN,XPIX,XLAT,XLON,XNU/XDR,XPSI/XDR,XRAD,
C    +  XLA/XDR,XTETA/XDR,XV/XDR,XRD,XALTC,XTH/XDR,XINCO/XDR,XPSIP/XDR
```

```
C550     FORMAT(1X//' IRD=',I3,'  XLIN,XPIX,XLAT,XLON=',2F8.1,2F12.6,/,
C     +  ' XNU=',F12.8,'  XPSI=',F13.8,'  XRAD=',F12.4,'  XLA=',F14.8/,
C     +  ' XTETA=',F13.8,'  XV=',F13.8,'  XRD=',F12.4,'  XALTC=',F10.6,/,
C     +  ' XTH=',F14.8,'  XINCO=',F12.6,'  XPSIP=',F12.6)
         XDLAM=DASIN(DSIN(XPSIP)*DSIN(XTH)/DCOS(XLATR)) ! DELTA LONG.:GP TO PI
         XA=(X90-XLATR+XTH)*0.5D0
         XB=(XPSIP-XDLAM)*0.5D0
         XC=(XPSIP+XDLAM)*0.5D0
         XLA=X90-2.0D0*DATAN(DCOS(XC)*DTAN(XA)/DCOS(XB))
         XLO=XLONR+XDLAM
C        WRITE(6,552)XA,XB,XC,XLA/XDR,XDLAM/XDR,XLO/XDR
C552     FORMAT(1X/' 22222: XA,XB,XC=',3F16.8,/,' XLA=',F14.8,
C     +  '  XDLAM=',F12.6,'  XLO=',F12.6)
         IRD=IRD+1
         IF(IRD.GE.6)GOTO 500
         IF(DABS(XLA-XLAO).GT.0.00004D0)THEN
           XLAO=XLA
           GOTO 540
         END IF
C
C        CALCULATE EQUATOR CROSSING LONG. AND TRAVEL TIME.
C
         XDLAM=DASIN(DTAN(XLA)*DTAN(XINC)) ! DELTA LONG - GP TO EQ. CROSSING
         XE1=DACOS(XDATW(4)+XRD*XCV/XDATW(1))  ! ECCENTRIC ANOMALY
         IF(XV.GT.XDATC(7))XE1=X2PI-XE1
         XT=XDATC(9)*(XE1-XDATW(4)*DSIN(XE1))-XDATC(6) ! TRAVEL TIME - EQUATOR
                                  ! TO GP
         IF(DSIGN(1.0D0,XTETA).NE.DSIGN(1.0D0,XT))XT=XT+
     +   DSIGN(1.0D0,XTETA)*XDATW(2)
         XLONE=XLO-XDLAM-XT*XDATC(4) ! EQ. CROSSING LONGITUDE(RAD).
         XT1=XT+XRN*(XLIN-1.0D0)*XDATW(6) ! STANDERDISED TRAVEL TIME IE.
                                      !  LINE 1 TO EQUATOR.
C        WRITE(6,572)XDLAM/XDR,XE1/XDR,XV/XDR,XT,XDATC(6),XTETA/XDR,
C     +  XLONE/XDR,XT1
C572     FORMAT(1X/' 44444: XDLAM=',F13.6,'  XE1=',F14.8,'  XV=',F14.8,/,
C     +  ' XT=',F14.6,'  XDATC(6)=',F14.6,'  XTETA=',F14.8,/,
C     +  ' XLONE=',F12.6,'  XT1=',F14.6,//)
C
         XSTOR(1,NUM)=XLONE
         XSTOR(2,NUM)=XT1
         WRITE(6,580)NUM,XLIN,XPIX,XLAT,XLON,XLONE/XDATC(8),XT1
580      FORMAT(1X,I3,F9.2,F10.2,2F12.5,2F13.5)
         XLA=DATAN(DTAN(XLA)/0.9933D0) ! GP. LAT FROM GEOCENTRIC TO GEODETIC
         IF(CINRP.NE.'Y')WRITE(5,600)XLA/XDATC(8),XLO/XDATC(8),
     +   XLONE/XDATC(8),XT,XT1
600      FORMAT(1X/' GP LAT AND LONG=',2F10.5,'  EQ. LONG=',F10.5,/,
     +   ' TRAVEL TIME : GP TO EQ=',F10.4,'  LINE 1 TO EQ.=',F10.4)
C
         GOTO 500                 ! GO INPUT NEXT DATA SET.
C
C ---------------------------------------------------------------------
C
C        CALCULATE MEAN AND STANDARD DEVIATION.
C
C        USE ALL DATA SETS ?
C
800      IF(NUM.EQ.0)GOTO 1160
         IF(NUM.EQ.1)GOTO 1100
810      WRITE(5,820)
820      FORMAT(1X//' MEAN AND ST. DEV CALCULATION :',/,
```

```
     +   ' DISCARD ANY OF THE DATA SETS ? (Y/N) :',$)
         READ(5,120)CHAR
         IND=0
         IF(CHAR.EQ.'N')GOTO 880
840      WRITE(5,860)
860      FORMAT(' ENTER NO. OF THE SET TO BE DISCARDED(0=NO MORE):',$)
         READ(5,*)I
         IF(I.LT.0.OR.I.GT.NUM)GOTO 840
         IF(I.EQ.0)GOTO 880
         IND=IND+1
         IDCSET(IND)=I
         GOTO 840
C
C        DO THE CALCULATION
C
880      IF(IND.NE.0)WRITE(6,900)(IDCSET(I),I=1,IND)
900      FORMAT(1X//' MEAN AND ST.DEV. AFTER DISCARDING NUMBER(S):',
     +   10I3,/,1X,20I3,/,1X,20I3)
         DO 1000 I=1,2
            XSUM=0.0D0
            XSUM2=0.0D0
            XN=0.0D0
            DO 920 J=1,NUM
               DO 910 K=1,IND
                  IF(J.EQ.IDCSET(K))GOTO 920
910            CONTINUE
               XN=XN+1
               XSUM=XSUM+XSTOR(I,J)
               XSUM2=XSUM2+XSTOR(I,J)**2.0
920         CONTINUE
            IF(XN.LT.1.0D0)GOTO 1160

            XDATC(I)=XSUM/XN
            IF(XN.LT.2.0D0)THEN
                XSTD=0.0D0
                GOTO 940
            END IF
            XSTD=DSQRT((XN*XSUM2-XSUM**2.0)/(XN*(XN-1.0D0)))
C
940         IF(I.EQ.1)WRITE(6,960)XN,XDATC(1)/XDATC(8),XSTD/XDATC(8)
            IF(I.EQ.2)WRITE(6,980)XN,XDATC(2),XSTD
960         FORMAT(1X/' EQ.LONG : N, MEAN, ST.DEV =',F5.0,2F10.4)
980         FORMAT(' TIME     :',18X,'=',F5.0,2F10.4)
1000     CONTINUE
         WRITE(5,1020)
1020     FORMAT(1X/' ANOTHER MEAN/ST.DEV. CALCULATION ? (Y/N) :',$)
         READ(5,120)CHAR
         IF(CHAR.NE.'N')GOTO 810
         GOTO 1120
C
C        EXIT
C
1100     XDATC(1)=XSTOR(1,1)
         XDATC(2)=XSTOR(2,1)
1120     WRITE(5,1140)XDATC(1)/XDATC(8),XDATC(2)
1140     FORMAT(1X//1X,F10.4,' DEG AND ',F10.4,' SEC ENTERED AS EQ.',
     +   'LONG AND TRAVEL TIME.')
         GOTO 1200
1160     WRITE(5,1180)
1180     FORMAT(1X//' EQ.LONG AND TRAVEL TIME NOT DEFINED !!!')
```

```
1200      IF(CINRP.EQ.'Y')CLOSE(1)
1300      RETURN
          END
```

```
C
C       SUBROUTINE NOAN2.FTN
C       A SUBPROGRAM TO NOANAV. THIS SUBROUTINE CALCULATES THE LAT. AND
C       LONG. FOR A GIVEN LINE/PIXEL COORDINATE.
C
C       J.J.AGENBAG   S.F.R.I    JUNE 1989.    IN FORTRAN 77
C
        SUBROUTINE NOAN2(XDATW,XDATC)
C
        IMPLICIT REAL*8(X-Z),CHARACTER*1(C)
        CHARACTER DATAFN*10
        DIMENSION XDATW(8),XDATC(10)
C
        XDR=XDATC(8)
C       WRITE(6,5)XDATW(1),XDATW(2)/60.0D0,XDATW(3)/XDR,XDATW(4),
C     +  XDATW(5)/XDR,XDATW(6),XDATW(7)/XDR,XDATC(1)/XDR,XDATC(2),
C     +  XDATC(3)/XDR,XDATC(4)/XDR,XDATC(5)/XDR,(XDATC(I),I=6,10)
C5      FORMAT(' 1. XDATW =:',/,' 1=',F12.6,'  2=',F12.6,'  3=',F12.6,/,
C     +  ' 4=',F12.8,'  5=',F12.6,'  6=',F10.8,'  7=',F12.8,/,
C     +  ' XDATC =:',/,' 1=',F12.6,' 2=',F13.6,'  3=',F12.6,/,
C     +  ' 4=',F12.6,'  5=',F12.6,'  6=',F12.6,'  7=',F15.8,/,
C     +  ' 8=',F15.8,'  9=',F15.8,'  10=',F15.8)
C
10      WRITE(5,20)
20      FORMAT(1X/' READ LINE/PIXEL COORDINATES FROM A FILE ? (Y/N):',$)
        READ(5,40)CINPD
40      FORMAT(A1)
        IF(CINPD.NE.'Y')GOTO 140
        WRITE(5,60)
60      FORMAT(' ENTER DATA FILE NAME( 10 CHAR.=MAX):',$)
        READ(5,80)DATAFN
80      FORMAT(A10)
        OPEN(1,FILE=DATAFN,STATUS='OLD',ERR=100)
        WRITE(5,90)
90      FORMAT(' NUMBER OF DATA SETS TO READ ? :',$)
        READ(5,*)NUMDS
        IF(NUMDS.LE.0)GOTO 10
        GOTO 140
100     WRITE(5,120)DATAFN
120     FORMAT(1X/' FILE ',A10,' NOT ACCESSABLE.')
        GOTO 10
C
C       PRINT HEADER AND CALCULATE SOME CONSTANTS
C
140     WRITE(6,160)
160     FORMAT(1X/' NUMBER',6X,'LINE',5X,'PIXEL',6X,'LATITUDE',
     +  '   LONGITUDE',/)
        XRN=DSIGN(1.0D0,XDATW(3))        ! 1= ASCENDING, -1= DESCENDING TRACK
        XC1=(1.0D0-XDATW(4)**2.0)*XDATW(1)
        X2PI=2.0D0*XDATC(7)
        XINC=(XDATW(3)-XRN*XDATC(7)/2.0D0) ! INCLINATION W OF N (RAD).
C
C
C ------------------ START CALCULATION CYCLE --------------------------
C
C       1.0 INPUT
C
        NUM=0              ! COUNTER FOR DATA SETS ENTERED.
200     IF(CINPD.EQ.'Y')THEN
```

```
            IF(NUM.EQ.NUMDS)GOTO 500      ! FINISHED
            READ(1,*)XLIN,XPIX            ! READ LINE AND PIXEL
          ELSE
            WRITE(5,220)NUM+1
220         FORMAT(1X/1X,I3,' : ENTER LINE AND PIXEL(0=NO MORE):',$)
            READ(5,*)XLIN,XPIX
            IF(XLIN.LT.1.0D-5.AND.XPIX.LE.1.0D-5)GOTO 500
        END IF
        NUM=NUM+1
C
C       2.0 FIND TRUE ANOMALY PERIFOCUS TO GP .
C
        XT=XDATC(2)-XRN*(XLIN-1.0D0)*XDATW(6) ! TRAVEL TIME EQUATOR TO GP
        XT1=XT+XDATC(6)           ! TRAVEL TIME PERIFOCUS TO GP
        IF(XT1.LT.0.0D0)XT1=XT1+XDATW(2)
        IF(XT1.GE.XDATW(2))XT1=XT1-XDATW(2)
        ZTF=XDATC(3)*XT1
C       FIND GP ECCENTRIC ANOMALY,XE1, BY ITERATION FROM THE EQUATION
C       XE1-E*SIN(XE1)=ZTF     E=ECCENTRICITY
C
        IRD=1              ! ITERATION STEP COUNTER
        XE1=ZTF            ! FIRST APPROXIMATION
240     ZE1=XE1-(XE1-XDATW(4)*DSIN(XE1)-ZTF)/(1.0D0-XDATW(4)*DCOS(XE1))
        IF(DABS(XE1-ZE1).LE.0.1D-5)GOTO 280 ! SUCCESS. END ITERATION
        IRD=IRD+1
        XE1=ZE1
        IF(IRD.LE.20)GOTO 240    ! NEXT ITERATION
        WRITE(5,260)
260     FORMAT(1X/' ERROR !!. ITERATION FOR ECCENTRIC ANOMALY FAILED.')
        GOTO 200
C
280     CONTINUE
C       WRITE(6,285)NUM,XLIN,XPIX,XT,ZTF/XDR,IRD,XE1/XDR
C285    FORMAT(1X//' 2. NUM=',I4,' XLIN=',F12.6,'  XPIX=',F12.6,/,
C     +  ' XT=',F16.8,'  ZTF=',F15.8,'  IRD=',I3,/,
C     +  ' XE1=',F15.8)
        XCV=(DCOS(XE1)-XDATW(4))/(1.0D0-XDATW(4)*DCOS(XE1)) ! COS TRUE ANOMALY
        XV=DACOS(XCV)    ! TRUE ANOMALY FOR GP.
        IF(XE1.GT.XDATC(7))THEN
          XV=X2PI-XV
          XCV=DCOS(XV)
        END IF
C
C       3.0 CALCULATE GP. LAT AND LONG.
C
        IF(XRN.GE.0.0D0)THEN     ! ASCENDING TRACK
           XTETA=XV-X2PI+XDATW(5) !GEOCENTRIC ANGLE EQUATOR TO GP.
           IF(XTETA.LT.-XDATC(7))XTETA=XTETA+X2PI
         ELSE                    ! DESCENDING TRACK.
           XTETA=XV-XDATW(7)+XDATW(5)
        END IF
        IF(XTETA.GE.XDATC(7))XTETA=XTETA-X2PI
        XLAT=DASIN(DCOS(XINC)*DSIN(XTETA))   ! GP. LATITUDE
        XINCO=DASIN(DSIN(XINC)/DCOS(XLAT))   ! TRACKLINE INCLINATION AT GP
        XDLAM=DASIN(DTAN(XINC)*DTAN(XLAT)) !LONG. DIFF. EQUAT. TO GP
        XLON=XDATC(1)+XDLAM+XT*XDATC(4)   ! GP. LONGITUDE
        XRD=XC1/(1.0D0+XDATW(4)*XCV) ! RADIAL DISTANCE, EARTH CENTRE TO SAT.
C
C       WRITE(6,290)XV/XDR,XTETA/XDR,XLAT/XDR,XINCO/XDR,XLAM/XDR,
C     +  XLON/XDR,XRD
```

```
C290     FORMAT(1X/' 3.  XV=',F12.6,'  XTETA=',F12.6,'  XLAT=',F12.6,/,
C     +  ' XINCO=',F12.6,'  XDLAM=',F15.8,'  XLON=',F12.6,'  XRD=',F12.6)
C
C        4.0 CALCULATE PIXEL LAT AND LONG. THE CALCULATION INVOLVES THE
C            EARTH RADIUS AT THE (UNKNOWN) PIXEL LATITUDE. FIRST CARRY
C            THROUGH THE CALCULATION USING THE RADIUS AT THE GP. THEN REPEAT
C            WITH THE DERIVED PIXEL LATITUDE.
C
         XNU=(XPIX-1024.5D0)*XDATW(7)     ! SCAN ANGLE : NADIR TO PIXEL
         XTHS=XNU
         XPSI=XDATC(5)    ! AZIMUTH WITH SCAN SCEW =CA. 90.01 DEG (RAD)
         XPSIP=XPSI-XINCO          ! TRUE BEARING , GP TO PIXEL
         IRD=1             ! FIRST CALCULATION
         XLA=XLAT          ! SET PIXEL LAT=GP LAT.
300      XRAD=6378.137D0*(1.0D0-.00335D0*(DSIN(XLA))**2.0) ! EARTH RADIUS(KM.)
         XALTC=XRD/XRAD
         XTH=DASIN(XALTC*DSIN(XTHS))-XTHS  ! GEOCENTRIC ANGLE : GP TO PIXEL
C        WRITE(6,310)IRD,XNU/XDR,XPSI/XDR,XPSIP/XDR,XLA/XDR,
C     +  XRAD,XALTC,XTH/XDR
C310     FORMAT(1X/' 4. IRD=',I3,'  XNU=',F12.6,'  XPSI=',F12.6,/,
C     +  ' XPSIP=',F12.6,'  XLA=',F12.6,/,
C     +  ' XRAD=',F15.6,'  XALTC=',F12.8,'  XTH=',F12.6)
         IF(IRD.GT.2)GOTO 340
         XLAP=DASIN(DCOS(XTH)*DSIN(XLAT)+DSIN(XTH)*DCOS(XLAT)*
     +   DCOS(XPSIP))     ! PIXEL LATITUDE
         XLA=XLAP
         IRD=IRD+1
         GOTO 300          ! REPEAT PIXEL LAT. CALCULATION WITH THIS LAT.
C
340      XDLAM=DASIN(DSIN(XPSIP)*DSIN(XTH)/DCOS(XLAP)) ! LONGITUDE DIFFERENCE
                           ! BETWEEN GP. AND PIXEL.
         XLOP=(XLON-XDLAM)/XDATC(8)       ! PIXEL LONG. (DEG)
         XLAP=DATAN(DTAN(XLAP)/0.9933D0) ! CONVERT PIXEL LAT. FROM GEOCENTRIC
                                          ! TO GEODETIC
         XLAP=XLAP/XDATC(8)               ! PIXEL LAT.  (DEG)
         XLAT=DATAN(DTAN(XLAT)/0.9933D0) ! G.P LAT FROM GEOCENTRIC TO GEODETIC
C        WRITE(6,350)XLAP,XDLAM/XDR,XLOP
C350     FORMAT(1X/' 5. XLAP=',F12.6,'  XDLAM=',F12.6,' XLOP=',F12.6,/)
C
         WRITE(6,400)NUM,XLIN,XPIX,XLAP,XLOP
400      FORMAT(1X,I4,F14.2,F10.2,F13.4,F11.4)
         IF(CINPD.NE.'Y')WRITE(5,420)XLAT/XDATC(8),XLON/XDATC(8),XLAP,
     +   XLOP
420      FORMAT('     GP LAT AND LONG =',F9.4,F12.4,/,
     +   ' PIXEL LAT AND LONG =',F9.4,F12.4)
         GOTO 200                  ! GET NEXT DATA SET.
C
C ---------------------- END CALCULATION CYCLE ----------------------
C
500      IF(CINPD.EQ.'Y')CLOSE(1)
600      RETURN
         END
```

```
C
C       SUBROUTINE NOAN3.FTN
C       A SUBPROGRAM TO NOANAV. THIS ROUTINE CALCULATES THE LINE AND PIXEL
C       NUMBER FOR A GIVEN LAT./LONG. POSITION.
C
C       J.J.AGENBAG     S.F.R.I     JUNE 1989.    IN FORTRAN 77
C
        SUBROUTINE NOAN3(XDATW,XDATC)
C
        IMPLICIT REAL*8(X-Z),CHARACTER*1(C)
        CHARACTER DATAFN*10
        DIMENSION XDATW(8),XDATC(10)
C
C
        XDR=XDATC(8)
C       WRITE(6,5)XDATW(1),XDATW(2)/60.0D0,XDATW(3)/XDR,XDATW(4),
C     +  XDATW(5)/XDR,XDATW(6),XDATW(7)/XDR,XDATC(1)/XDR,XDATC(2),
C     +  XDATC(3)/XDR,XDATC(4)/XDR,XDATC(5)/XDR,(XDATC(I),I=6,10)
C5      FORMAT(' 1. XDATW =:',/,' 1=',F12.6,'  2=',F12.6,'  3=',F12.6,/,
C     +  ' 4=',F12.8,'  5=',F12.6,'  6=',F10.8,'  7=',F12.8,/,
C     +  ' XDATC =:',/,' 1=',F12.6,' 2=',F13.6,'  3=',F12.6,/,
C     +  ' 4=',F12.6,'  5=',F12.6,'  6=',F12.6,'  7=',F15.8,/,
C     +  ' 8=',F15.8,'  9=',F15.8,'  10=',F15.8)
C
10      WRITE(5,20)
20      FORMAT(1X/' READ LAT./LONG. COORDINATES FROM A FILE ? (Y/N):',$)
        READ(5,40)CINPD
40      FORMAT(A1)
        IF(CINPD.NE.'Y')GOTO 140
        XSL=0.0D0
        XSP=0.0D0
        WRITE(5,60)
60      FORMAT(' ENTER DATA FILE NAME( 10 CHAR.=MAX):',$)
        READ(5,80)DATAFN
80      FORMAT(A10)
        OPEN(1,FILE=DATAFN,STATUS='OLD',ERR=100)
        WRITE(5,90)
90      FORMAT(' NUMBER OF DATA SETS TO READ ? :',$)
        READ(5,*)NUMDS
        IF(NUMDS.LE.0)GOTO 10
        GOTO 140
100     WRITE(5,120)DATAFN
120     FORMAT(1X/' FILE ',A10,' NOT ACCESSABLE.')
        GOTO 10
C
C       PRINT HEADER AND CALCULATE SOME CONSTANTS
C
140     WRITE(6,160)
160     FORMAT(1X/' NO.      LAT',8X,'LONG      C LINE  C PIXEL    LINE',
     +  '    PIXEL   L.ERR P.ERR',/)
        XRN=DSIGN(1.0D0,XDATW(3))          ! 1=ASCENDING AND -1=DESCENDING TRACK.
        XC1=(1.0D0-XDATW(4)**2.0)*XDATW(1)
        XINC=(XDATW(3)-XRN*XDATC(7)/2.0D0) ! INCLINATION W OF N (RAD).
        X2PI=2.0D0*XDATC(7)                ! 2*PI RADIANS
        X90=0.5D0*XDATC(7)                 ! PI/2 RADIANS
C
C
C ------------------ START CALCULATION CYCLE -------------------------
C
```

```
C       1.0 INPUT
C
        NUM=0              ! COUNTER FOR DATA SETS ENTERED.
        XNUM=0.0D0
200     IF(CINPD.EQ.'Y')THEN
           IF(NUM.EQ.NUMDS)GOTO 1000     ! FINISHED
           READ(1,*)XL,XP,XLAT,XLON      ! READ LAT AND LONG
         ELSE
          WRITE(5,220)NUM+1
220       FORMAT(1X/1X,I3,' : ENTER LAT AND LONG(S  AND E =+,'
     +     ' 0=END)',$)
          READ(5,*)XLAT,XLON
          IF(XLAT.LT.1.0D-5.AND.XLON.LE.1.0D-5)GOTO 1000
        END IF
        NUM=NUM+1
        XDR=XDATC(8)
        XLAP=-XLAT*XDR
        XLAP=DATAN(0.9933D0*DTAN(XLAP)) ! CONVERT PIXEL LAT. FROM GEODETIC
                                        ! TO GEOCENTRIC.
        XLOP=-XLON*XDR
        XRAD=6378.137D0*(1.0D0-.00335D0*(DSIN(XLAP))**2.0) ! EARTH RADIUS
                                        ! AT GIVEN LATITUDE (KM)
C
        ITER=1
        XPL=0.0D0                  ! SET INITIAL PIXEL VALUE
        XPH=2049.0D0               ! FOR ITERATION
500     XPIX=(XPL+XPH)*0.5D0
C
C       ITERATE TO FIND G.P LATITUDE. INITIALISE TO PIXEL LATITUDE. TERMINATI
C       ITERATION WHEN DIFFERENCE BETWEEN TWO SUCCESSIVE RESULTS(=0.00004 RAI
C
        IRD=1              ! FIRST CALCULATION
        XNU=(XPIX-1024.5D0)*XDATW(7)     ! SCAN ANGLE TO PIXEL (RAD)
        XTHS=XNU
        XPSI=XDATC(5)    ! RELATIVE BEARING TO PIXEL =CA 90.01DEG IN RAD
        XRAD=6378.137D0*(1.0D0-.00335D0*(DSIN(XLAP))**2.0) ! EARTH RADIUS
                                                  ! AT PIXEL
        XLA=XLAP           ! INITIALISE GP. LAT TO PIXEL LAT
        XLAO=XLA           ! SAVE PREVIOUS RESULT FOR ITERATION CHECK.
540     XTETA=XRN*DASIN(DSIN(XLA)/DCOS(XINC))  ! ANGULAR DISTANCE EQ. TO XLA
        IF(XRN.GE.0.0D0)THEN               ! ASCENDING TRACK
           XV=X2PI+XTETA-XDATW(5) ! TRUE ANOMALY=ANG. PERIFOCUS TO XLA
           IF(XV.GE.X2PI)XV=XV-X2PI
         ELSE                      ! DESCENDING TRACK
          XV=XDATC(7)+XTETA-XDATW(5)
        END IF
        IF(XV.LT.0.0D0)XV=X2PI+XV
        XCV=DCOS(XV)
        XINCO=DASIN(DSIN(XINC)/DCOS(XLA))
        XPSIP=XPSI-XINCO
        XRD=XC1/(1.0D0+XDATW(4)*XCV)  ! GEOCENTRIC DIST. TO SATELLITE
                                      ! AT THIS LATITUDE.
        XALTC=XRD/XRAD             ! ALTITUDE FACTOR
        XTH=DASIN(XALTC*DSIN(XTHS))-XTHS  ! GEOCENTRIC ANGLE - GP TO PIXEL
        XDLAM=DASIN(DSIN(XPSIP)*DSIN(XTH)/DCOS(XLAP))   ! DELTA LONGITUDE
        XA=(X90-XLAP+XTH)*0.5D0
        XB=(XPSIP-XDLAM)*0.5D0
        XC=(XPSIP+XDLAM)*0.5D0
        XLA=X90-2.0D0*DATAN(DCOS(XC)*DTAN(XA)/DCOS(XB))
        IF(DABS(XLA-XLAO).GT.0.00004D0)THEN
```

```
          XLAO=XLA
          IRD=IRD+1
          GOTO 540
        END IF
C
C       CALCULATE GP. LONGITUDE
C
        XLO=XLOP+XDLAM              ! GP. LONGITUDE (RAD)
C
C       CALCULATE EQUATOR CROSSING LONG. AND TRAVEL TIME.
C
        XDLAM=DASIN(DTAN(XLA)*DTAN(XINC)) ! DELTA LONG - GP TO EQ. CROSSING
        XE1=DACOS(XDATW(4)+XRD*XCV/XDATW(1))  ! ECCENTRIC ANOMALY
        IF(XV.GT.XDATC(7))XE1=X2PI-XE1
        XT=XDATC(9)*(XE1-XDATW(4)*DSIN(XE1))-XDATC(6) ! TRAVEL TIME - EQUATOR
                                                      ! TO GP
        IF(DSIGN(1.0D0,XTETA).NE.DSIGN(1.0D0,XT))XT=XT+
     +  DSIGN(1.0D0,XTETA)*XDATW(2)
        XLONE=XLO-XDLAM-XT*XDATC(4) ! EQ. CROSSING LONGITUDE(RAD).
C
C       WRITE(6,580)ITER,XPIX,XLA/XDR,XLO/XDR,XT,XLONE/XDR,XDATC(1)/XDR
C580    FORMAT(' ITERATION NO',I3,' PIXEL=',F10.4,' GP LAT AND LONG=',
C    +  2F10.4,/,' XT=',F12.5,'  XLONE=',F10.5,'  TRUE NOD LONG=',F10.5)
C
C       CHECK AGAINST TRUE NODAL LONGITUDE.
C
        XDIF=XLONE-XDATC(1)
        IF(DABS(XDIF).LT.1.74D-5)GOTO 600
        ITER=ITER+1
        IF(ITER.GT.20)THEN
            XLIN=9999.9D0
            XPIX=9999.9D0
            GOTO 800
          ELSE
            IF(XDIF.GT.0.0D0)XPH=XPIX
            IF(XDIF.LT.0.0D0)XPL=XPIX
            GOTO 500
        END IF
C
C600    XPIX=1024.5D0+XNU/XDATW(7)       ! THE PIXEL NUMBER
600     XLIN=1.0D0+XRN*(XDATC(2)-XT)/XDATW(6)   ! THE LINE NUMBER
C
C
        IF(CINPD.EQ.'Y')THEN
800       WRITE(6,820)NUM,-XLAT,-XLON,XLIN,XPIX,XL,XP,XL-XLIN,XP-XPIX
820       FORMAT(1X,I3,2F11.5,2X,2F8.2,2X,2F7.1,2X,2F5.1)
          IF(XPIX.LE.5000.0D0)THEN
            XSL=XSL+(XL-XLIN)**2.0
            XSP=XSP+(XP-XPIX)**2.0
            XNUM=XNUM+1.0D0
          END IF
         ELSE
          XLA=DATAN(DTAN(XLA)/0.9933D0) ! CONVERT G.P LAT FROM GEOCENTRIC TO
                                        ! GEODETIC
          WRITE(5,840)XLA/XDR,XLO/XDR,XLIN,XPIX
840       FORMAT('  GP. LAT AND LONG =',F9.4,F12.4,/,
     +     '  LINE =',F7.1,'  PIXEL =',F7.1)
        END IF
        GOTO 200
1000    IF(CINPD.EQ.'Y')THEN
```

```
          CLOSE(1)
          XSL=DSQRT(XSL/XNUM)
          XSP=DSQRT(XSP/XNUM)
          WRITE(6,1020)XNUM,XSL,XSP
1020      FORMAT(1X/' NUMBER OF DATA SETS =',F5.0,/,
     +     ' RMS LINE ERROR =',F7.3,'   RMS PIXEL ERROR =',F7.3)
        END IF
        RETURN
        END
```

```
C
C       SUBROUTINE NOAN4.FTN
C       A SUBPROGRAM TO NOANAV. THIS ROUTINE INPUTS ORBITAL ELEMENTS AND
C       SCANNER DATA.
C
C       J.J.AGENBAG  S.F.R.I     JUNE 1989 .    IN FORTRAN 77
C
        SUBROUTINE NOAN4(XDATI)
C
        IMPLICIT REAL*8(X-Z),CHARACTER*1(C)
        DIMENSION XDATI(8)
C
        INUM=1
30      CONTINUE
C
        WRITE(5,40)(XDATI(I),I=1,7)
40      FORMAT(1X//' ORBIT AND SCANNER DATA ARE:',/
     + ' 1. SEMI-MAJOR AXIS (KM)',23X,'=',F11.5,/,
     + ' 2. NODAL PERIOD (MINUTES)',21X,'=',F11.6,/,
     + ' 3. INCLINATION ANGLE (WEST WARD FROM EQUATOR) =',F12.8,/,
     + '     ( A NEGATIVE INCLINATION INDICATES A DESCENDING TRACK )',/,
     + ' 4. ECCENTRICITY OF ORBIT',22X,'=',F11.9,/,
     + ' 5. ARGUMENT OF PERIGEE (DEG)',18X,'=',F11.6,/,
     + '     (NOTE : 4 & 5 MAY BE = 0.0  IE. CIRCULAR ORBIT)',/,
     + ' 6. SCANNER SCAN RATE (SECONDS PER SCAN)        =',F11.8,/,
     + ' 7. SCANNER DIGITIZATION STEP ANGLE ( DEG )     =',F14.10,/,
     + '     ( SET TO ZERO FOR CALCULATION FROM SCAN RATE)',/)
C
60      WRITE(5,80)
80      FORMAT(' CHANGE ANY OF THIS ? (Y/N) :',$)
        READ(5,100)CHAR
100     FORMAT(A1)
        IF(CHAR.EQ.'N')GOTO 140
110     WRITE(5,120)
120     FORMAT(1X/' ENTER ITEM NUMBER(0=NO MORE) AND NEW VALUE :',$)
        READ(5,*)INUM,XVALUE
        IF(INUM.EQ.0.OR.INUM.GT.7)THEN
            GOTO 30
          ELSE
            XDATI(INUM)=XVALUE
            GOTO 110
        END IF
C
C
140     RETURN
        END
```

```
C
C       SUBROUTINE NOAN5.FTN
C       A SUBPROGRAM TO NOANAV. THIS SUBROUTINE CALCULATE CERTAIN CONSTANTS.
C
C       J.J.AGENBAG     S.F.R.I    JUNE 1989.     IN FORTRAN 77
C
        SUBROUTINE NOAN5(XDATI,XDATW,XDATC)
C
        IMPLICIT REAL*8(X-Z),CHARACTER*1(C)
        DIMENSION XDATI(8),XDATW(8),XDATC(10)
C
        XPI=4.0D0*DATAN(1.0D0)   ! PI
        X2PI=2.0D0*XPI           ! 2PI
        XDR=XPI/180.0D0          ! DEGREE TO RADIAN CONVERSION FACTOR.
        IF(XDATI(7).LT.0.1D-5)THEN
           XDATI(7)=360.0D0/(XDATI(6)*39936) ! CALCULATE STEP ANGLE FROM
                                             ! SCAN RATE.
        END IF
C
        XDATW(1)=XDATI(1)
        XDATW(2)=XDATI(2)*60.0D0
        XDATW(3)=XDATI(3)*XDR
        XDATW(4)=XDATI(4)
        XDATW(5)=XDATI(5)*XDR
        XDATW(6)=XDATI(6)
        XDATW(7)=XDATI(7)*XDR
C
C       WRITE(5,20)
C20     FORMAT(1X//' NOAN5 : REMOVE SCAN SCEW ? (Y/N):',$)
C       READ(5,40)CYN
C40     FORMAT(A1)
C
C
C       SOME CONSTANTS
C
        XRN=DSIGN(1.0D0,XDATI(3))        ! 1=ASCENDING TRACK, -1=DESCENDING
        XPER=XDATI(2)*60.0D0     ! NODAL PERIOD IN SECONDS
        XINCR=(XDATI(3)-XRN*90.0D0)*XDR ! INCLINATION WEST OF NORTH(RAD)
        XRSCAN=(2048.0D0-1024.5D0)*XDATI(7)*XDR ! MAX. SCAN ANGLE(RAD)
                                 ! FOR SCAN SCEW CALCULATION.
        XR=XPER/(2.0D0*XPI)
        XROTR=2.0D0*XPI/86164.09D0 ! EARTH ROTATION RATE(WESTWARD,RAD/SEC)
        XPRER=-1.5D0*0.00108263D0*(6371.0D0**2.0)*DSQRT(398603.0D0)*
     +  ((XDATI(1))**-3.5)*DCOS(XDATI(3)*XDR) ! ORBIT PRECESSION (RAD/SEC)
        XRM=XROTR-XPRER          ! COMBINED MOTION. WESTWARD
        XC1=1.0D0-XDATI(4)**2.0
        XC2=XDATI(1)*XC1
        XC3=XDATI(1)*DSQRT(XC1)
        IF(XRN.GE.0.0D0)THEN             ! ASCENDING
           XV0=X2PI-XDATW(5)   ! TRUE ANOMALY. NODAL POINT TO P.F
          ELSE                           ! DESCENDING
           XV0=XPI-XDATW(5)
           IF(XV0.LT.0.0D0)XV0=XV0+X2PI
        END IF
        XCV0=DCOS(XV0)                ! COSINE OF TRUE ANOMALY.
        XRD=XC2/(1.0D0+XDATI(4)*XCV0) ! SATELLITE GEOCENTRIC DISTANCE AT EQUAT.
        XE0=DACOS(XDATI(4)+XRD*XCV0/XDATI(1)) ! ECCENTRIC ANOMALY CORRESPON-
                                              ! TO XV0
        IF(XV0.GT.XPI)XE0=X2PI-XE0
```

```
      XT0=XR*(XE0-XDATI(4)*DSIN(XE0))  ! TRAVEL TIME BETWEEN PERIFOCUS AND I
C
C     WRITE(6,60)XC1,XC2,XDATI(4),XV0/XDR,XCV0,XRD,XDATI(1),
C    +  XE0/XDR,XR*XDR,XT0,XR,XE0,DSIN(XE0)
C60   FORMAT(1X/' NOAN5 :',/,
C    +  ' XC1=',F12.8,'  XC2=',F13.7,'  XDATI(4)=',F12.7,/,
C    +  ' XV0=',F12.7,'  XCV0=',F10.7,'  XRD=',F13.7,'  XDATI(1)=',
C    +  F13.7,/,
C    +  ' XE0=',F12.7,'  XR=',F13.7,'  XT0=',F12.6,/,
C    +  ' XR AND XE0 IN RAD.=',2F14.7,'  SIN(XE0)=',F10.7,/)
C
C     SCAN SCEW CALCULATION : IF PIXEL NO 2048 IS AT LATITUDE 28.5 S THEN
C     THE SATELLITE GP IS AT APPROX. 32 DEG SOUTH. USE A CONSTANT SCAN
C     SCEW CALCULATED FOR THESE LATITUDES.
C     NOTE : THIS ASSUMPTION ONLY AFFECTS THE CALCULATION OF THE EARTH
C     RADIUS AND RADIAL DISTANCE TO THE SATELLITE IE TO GET THE ALTITUDE
C     FACTOR XALTCG
C
      XDT=XRSCAN*XDATI(6)/XPER          ! A FACTOR IN SCAN SCEW
      XRADG=6378.137D0*(1.0D0-.00335D0*(DSIN(28.5D0*XDR))**2.0) ! EARTH
                  ! RADIUS AT 28.5 LAT. FOR SCAN SCEW CALCULATION.
      XTETA=XRN*DASIN(DSIN(-32.0D0*XDR)/DCOS(XINCR)) ! ANGULAR DIST. TO EQ.
                  ! FROM GROUND POINT LATITUDE 32.0 DEG SOUTH
      IF(XRN.GE.0.0D0)THEN     ! ASCENDING TRACK
         XV=X2PI-XDATW(5)+XTETA  ! TRUE ANOMALY . PERIFOCUS TO 32 DEG S
         IF(XV.GE.X2PI)XV=XV-X2PI
        ELSE
         XV=XPI-XDATW(5)+XTETA   ! DESCENDING TRACK
      END IF
      IF(XV.LT.0.0D0)XV=XV+X2PI
      XRD1=XC2/(1.0D0+XDATI(4)*DCOS(XV))      !RADIAL DISTANCE
      XALTCG=XRD1/XRADG          ! ALTITUDE FACTOR FOR SCAN SCEW.
      XCSCAN=DASIN(XALTCG*DSIN(XRSCAN))-XRSCAN ! EARTH CENTRE ANGLE
                  ! CORRESPONDING TO A SCAN ANGLE TO PIX 2048
      XGAM=XPI-DACOS(XDT/XCSCAN)        ! AZIMUTH WITH SCAN SCEW
C     IF(CYN.EQ.'Y')XGAM=XPI/2.0D0      ! SET REMOVE SCAN SCEW EFFECT.
C
C     WRITE(6,100)XINCR/XDR,XRSCAN/XDR,XROTR/XDR,XPRER/XDR,XV0/XDR,
C    +  XRD,XE0/XDR,XT0,XDT/XDR,XRADG,XTETA/XDR,XV/XDR,XRD1,XCSCAN/XDR,
C    +  XGAM/XDR
C100  FORMAT(1X///' XINCR=',F12.7,'  XRSCAN=',F12.7,'  XROTR=',F12.7/,
C    +  ' XPRER=',F12.7,'  XV0=',F12.7,'  XRD=',F12.6,'  XE0=',F12.7,/,
C    +  ' XT0=',F12.5,'  XDT=',F12.7,'  XRADG=',F12.6,'  XTETA=',F12.7/,
C    +  ' XV=',F12.7,'  XRD1=',F12.6,'  XCSCAN=',F12.7,' XGAM=',F12.7,/)


      XDATC(1)=0.0D0
      XDATC(2)=0.0D0
      XDATC(3)=2.0D0*XPI/XPER
      XDATC(4)=XRM
      XDATC(5)=XGAM
      XDATC(6)=XT0
      XDATC(7)=XPI
      XDATC(8)=XDR
      XDATC(9)=XR

      RETURN
      END
```

```
C
C       SUBROUTINE NOAN6.FTN
C       A SUBROUTINE FOR PROGRAM 'NOANAV'. NOAN6 OBTAIN SEMI-MAJOR AXIS,
C       NODAL PERIOD, INCLINATION, ECCENTRICITY AND ARGUMENT OF PERIGEE
C       DATA BY INTERPOLATION (BY TIME) BETWEEN TWO EPOCHS TAKEN FROM THE
C       TBUS BULLETINS.
C
C       NB !! DESCENDING (SOUTH BOUND) AND ASCENDING (NORTH BOUND) TRACKS
C       ARE DISTINGUISED BETWEEN BY MEANS OF THE INCLINATION ANGLE. THE
C       PROGRAM THUS CHANGES THE INCLINATION TO NEGATIVE IN THE CASE OF
C       IMAGES FROM THE DESCENDING TRACK.
C
C       PROGRAM BY J.J.AGENBAG  S.F.R.I   JUNE 1989.  IN FORTRAN 77
C
C
        SUBROUTINE NOAN6(XDATI)
C
        IMPLICIT REAL*8(X-Z),CHARACTER*1(C)
        DIMENSION XDATI(8),DAYS(12),RDATE(7,3),RELEM(6,3)
        CHARACTER DATE(2)*15,PERM(2)*9,NEAI(5,2)*8,CBUF*15
        DATA DAYS/31.,28.,31.,30.,31.,30.,31.,31.,30.,31.,30.,31./
C
C       1.0 ENTER DATA
C
        WRITE(5,100)
100     FORMAT(1X/' NOAN6 : OBTAIN ORBITAL ELEMENTS BY TBUS',
     +  ' INTERPOLATION :',/,
     +  ' DATA ITEMS FROM PART IV OF TWO TBUS BULLETINS ARE ENTERED',/,
     +  ' THE DATA ARE ENTERED AS BLOCKS OF CHARACTERS EXACTLY AS THEY',
     +  ' APPEAR IN THE BULLETINS -',/,
     +  ' IE. WITHOUT ANY DECIMAL POINTS, EXTRA BLANKS ETC. :',//)
C
120     WRITE(5,140)
140     FORMAT(' DO YOU WISH TO APPLY THESE DATA TO AN IMAGE FROM AN',/,
     +  ' ASCENDING TRACK(A) OR A DESCENDING TRACK (D) ? (A/D):',$)
        READ(5,160)CTR
160     FORMAT(A1)
        IF(CTR.EQ.'D')THEN
           WRITE(5,180)
180        FORMAT(1X/' PLEASE NOTE : TO COMPLY WITH THE REQUIREMENTS ',
     +     'OF NOANAV, THE INCLINATION',/,
     +     ' ANGLE FOR DESCENDING TRACKS WILL BE CHANGED TO NEGATIVE.'/)
          ELSE IF(CTR.EQ.'A')THEN
           WRITE(5,185)
185        FORMAT(1X/)
          ELSE
           GOTO 120
        END IF
C
        IFLAG=0
        I=1                ! BULLETIN NO.1
200     IF(I.EQ.1)WRITE(5,220)
        IF(I.NE.1)WRITE(5,240)
220     FORMAT(' FOR BULLETIN NO.1 ENTER THE FOLLOWING :')
240     FORMAT(1X/' FOR BULLETIN NO.2 ENTER THE FOLLOWING :')
C
C       1.1 ENTER DATE/TIME BLOCK
C
260     WRITE(5,280)
```

```
280     FORMAT(' DATE/TIME: LINE 1, BLOCK 5 (15 CHAR.):',$)
        READ(5,300)DATE(I)
300     FORMAT(A15)
        WRITE(CBUF,300)DATE(I)
        READ(CBUF,320)(RDATE(J,I),J=1,6)
320     FORMAT(5F2.0,F5.3)
        ASSIGN 340 TO IGO
        GOTO 2000                ! CONVERT DATE/TIME TO A DAY COUNT.
340     IF(IFLAG.NE.0)GOTO 820   ! FOR DATA CORRECTION IFLAG IS SET=1
C
C       1.2 ENTER NODAL PERIOD.
C
360     WRITE(5,380)
380     FORMAT(1X/' NODAL PERIOD: LINE 2, BLOCK 2 (8 CHAR) :',$)
        READ(5,400)NEAI(1,I)
400     FORMAT(A8)
        WRITE(CBUF,400)NEAI(1,I)
        READ(CBUF,420)RELEM(1,I)
420     FORMAT(F8.4)
        IF(IFLAG.NE.0)GOTO 820
C
C       1.3 ENTER ECCENTRICITY
C
440     WRITE(5,460)
460     FORMAT(1X/' ECCENTRICITY: LINE 2, BLOCK 3 (8CHAR):',$)
        READ(5,400)NEAI(2,I)
        WRITE(CBUF,400)NEAI(2,I)
        READ(CBUF,480)RELEM(2,I)
480     FORMAT(F8.8)
        IF(IFLAG.NE.0)GOTO 820
C
C       1.4 ENTER ARGUMENT OF PERIGEE ( ANGLE BETWEEN EQUATOR AND PERIFOCUS)
C
500     WRITE(5,520)
520     FORMAT(1X/' ARG. OF PERIGEE: LINE 2, BLOCK 4 (8CHAR) :',$)
        READ(5,400)NEAI(3,I)
        WRITE(CBUF,400)NEAI(3,I)
        READ(CBUF,540)RELEM(3,I)
540     FORMAT(F8.5)
        IF(IFLAG.NE.0)GOTO 820
C
C       1.5 ENTER OREIT INCLINATION.
C
560     WRITE(5,580)
580     FORMAT(1X/' INCLINATION: LINE 2, BLOCK 6 (8CHAR) :',$)
        READ(5,400)NEAI(4,I)
        WRITE(CBUF,400)NEAI(4,I)
        READ(CBUF,540)RELEM(4,I)
        IF(IFLAG.NE.0)GOTO 820
C
C       1.6 ENTER SEMI-MAJOR AXIS
C
600     WRITE(5,620)
620     FORMAT(1X/' SEMI-MAJOR AXIS: LINE 3, BLOCK 2 (8CHAR) :',$)
        READ(5,400)NEAI(5,I)
        WRITE(CBUF,400)NEAI(5,I)
        READ(CBUF,640)RELEM(5,I)
640     FORMAT(F8.3)
        IF(IFLAG.NE.0)GOTO 820
C
```

```
C       1.7 ENTER MOTION OF PERIGEE
C
660     WRITE(5,680)
680     FORMAT(1X/' PERIGEE MOTION: LINE 5, BLOCK 2 (9CHAR) :',$)
        READ(5,700)PERM(I)
700     FORMAT(A9)
        WRITE(CBUF,400)(PERM(I)(2:9))
        READ(CBUF,540)RELEM(6,I)
        IF(PERM(I)(1:1).NE.'P')RELEM(6,I)=-1.0*RELEM(6,I)
        IF(IFLAG.NE.0)GOTO 820
C -----------------------------------------------
        IF(I.EQ.2)GOTO 740
        I=I+1
        GOTO 200
C -----------------------------------------------
C
C       2.0 DISPLAY AND CORRECT THE ENTRIES.
C
740     WRITE(5,760)(DATE(I),I=1,2),((NEAI(J,I),I=1,2),J=1,5),
     +  (PERM(I),I=1,2)
760     FORMAT(1X//' ITEM  LINE  BLOCK  CHAR',6X,'CONTENTS',9X,
     +  'BULLETIN 1',8X,'BULLETIN 2',//,
     +  1X,'  1.',6X,'1      5    15     DATE AND TIME',5X,A15,3X,A15,/,
     +  1X,'  2.',6X,'2      2     8     NODAL PERIOD',6X,A8,10X,A8,/,
     +  1X,'  3.',6X,'2      3     8     ECCENTRICITY',6X,A8,10X,A8,/,
     +  1X,'  4.',6X,'2      4     8     ARG. OF PERIGEE   ',A8,10X,A8,/,
     +  1X,'  5.',6X,'2      6     8     INCLINATION',7X,A8,10X,A8,/,
     +  1X,'  6.',6X,'3      2     8     SEMI-MAJOR AXIS   ',A8,10X,A8,/,
     +  1X,'  7.',6X,'5      2     9     PERIGEE MOTION    ',A9,9X,A9,/)
        WRITE(5,780)
780     FORMAT(' CHANGE ANY OF THIS ? (Y/N) :',$)
        READ(5,800)CHAR
800     FORMAT(A1)
        IF(CHAR.EQ.'N')GOTO 900
820     WRITE(5,840)
840     FORMAT(' ENTER ITEM AND BULLETIN NO.(0,0=NO MORE):',$)
        READ(5,*)ITEM,I
        IF(ITEM.EQ.0.AND.I.EQ.0)GOTO 740
        IF(ITEM.LT.1.OR.ITEM.GT.7.OR.I.LT.1.OR.I.GT.2)GOTO 820
        IFLAG=1
        GOTO(260,360,440,500,560,600,660),ITEM
C
C       3.0 ENTER IMAGE DATE AND PERFORM THE INTERPOLATION.
C
900     WRITE(5,920)
920     FORMAT(1X//' ENTER IMAGE DATE/TIME(Y,M,D,HR,MIN,SEC):',$)
        READ(5,*)(RDATE(I,3),I=1,6)
        I=3
        ASSIGN 940 TO IGO
        GOTO 2000          ! CONVERT DATE/TIME TO A DAY COUNT.
C
C       3.1 INTERPOLATION
C
940     TDIF1=RDATE(7,2)-RDATE(7,1)  ! TIME DIF. BETWEEN EPOCHS (BULLETINS)
        TDIF2=RDATE(7,3)-RDATE(7,1)  !    ,,   ,,  EPOCH 1 TO IMAGE TIME
        TDIF3=RDATE(7,3)-RDATE(7,2)  !    ,,   ,,  EPOCH 2 TO IMAGE TIME
        TFAC=TDIF2/TDIF1
        DO 960 J=1,6
           RELEM(J,3)=RELEM(J,1)+TFAC*(RELEM(J,2)-RELEM(J,1))
960     CONTINUE
```

```
C
C       USE THE MEAN PERIGEE MOTION OF IMAGE TIME AND THE EPOCH NEAREST
C       - IN TIME- TO THE IMAGE TIME , TO CALCULATE THE ARG. OF PERIGEE.
C
        RELEM(3,3)=RELEM(3,1)+((RELEM(6,1)+RELEM(6,3))/2.0)*TDIF2
        IF(ABS(TDIF3).LT.ABS(TDIF2))THEN  ! EPOCH 2 NEARER IN TIME
          RELEM(3,3)=RELEM(3,2)+((RELEM(6,2)+RELEM(6,3))/2.0)*TDIF3
        END IF
C
C       4.0 OUTPUT
C
        WRITE(6,760)(DATE(I),I=1,2),((NEAI(J,I),I=1,2),J=1,5),
     +  (PERM(I),I=1,2)
C       WRITE(6,980)((RDATE(I,J),I=1,7),J=1,3),((RELEM(I,J),J=1,3),
C    +  I=1,6),TDIF1,TDIF2,TDIF3,TFAC
980     FORMAT(1X/' DATE 1=',5F3.0,F6.3,F15.8/' DATE 2=',5F3.0,F6.3,
     +  F15.8/' DATE 3=',5F3.0,F6.3,F15.8,6(/,1X,3F16.8),/,
     +  ' TDIF1=',F13.8,'  TDIF2=',F13.8,'  TDIF3=',F13.8,
     +  ' TFAC =',F13.8,/)
C
        XDATI(1)=RELEM(5,3)
        XDATI(2)=RELEM(1,3)
        XDATI(3)=RELEM(4,3)
        IF(CTR.EQ.'D')XDATI(3)=-1.0D0*XDATI(3) ! DESCENDING TRACK. CHANGE
                                               ! INCLINATION TO NEGATIVE.
        XDATI(4)=RELEM(2,3)
        XDATI(5)=RELEM(3,3)
        WRITE(5,1000)(INT(RDATE(I,3)),I=1,5),RDATE(6,3),(XDATI(J),
     +  J=1,5)
        WRITE(6,1000)(INT(RDATE(I,3)),I=1,5),RDATE(6,3),(XDATI(J),
     +  J=1,5)
1000    FORMAT(1X//' ORBITAL ELEMENTS FOR THE IMAGE TIME ',
     +  3I3,I4,':',I2,':',F7.3,'   :',/,
     +  ' SEMI-MAJOR AXIS (KM)',26X,'=',F12.5,/,
     +  ' NODAL PERIOD (MINUTES)',24X,'=',F13.6,/,
     +  ' INCLINATION ANGLE (DEG WESTWARD FROM EQUATOR) =',F15.8,/,
     +  ' ( A NEGATIVE INCLINATION INDICATES A DESCENDING TRACK )',/,
     +  ' ECCENTRICITY OF ORBIT',25X,'=',F16.9,/,
     +  ' ARGUMENT OF PERIGEE (DEG)',2X,'=',F14.7,/)
        WRITE(5,1020)
1020    FORMAT(1X//' ANOTHER IMAGE TIME ? (Y/N):',$)
        READ(5,800)CHAR
        IF(CHAR.NE.'N')GOTO 900
        GOTO 2100
C
C -------------------------------------------------------------------
C
C       INTERNAL ROUTINE TO CONVERT DATE/TIME TO A DAY COUNT.
C
2000    RDATE(7,I)=0.0
        IDE=INT(RDATE(2,I))-1
        IF(IDE.EQ.0)GOTO 2040
        DO 2020 ID=1,IDE
           RDATE(7,I)=RDATE(7,I)+DAYS(ID)
           IF(ID.EQ.2)THEN      ! CHECK FOR LEAP YEARS.
              IF(RDATE(1,I).EQ.(FLOAT(INT(RDATE(1,I)/4.0))*4.0))
     +        RDATE(7,I)=RDATE(7,I)+1.0
           END IF
2020    CONTINUE
2040    RDATE(7,I)=RDATE(7,I)+RDATE(3,I)+RDATE(4,I)/24.0+RDATE(5,I)/
```

```
     +   (24.0*60.0)+RDATE(6,I)/(24.0*60.0*60.0)
         IF(I.EQ.1)GOTO 2060
         IF(RDATE(1,1).EQ.RDATE(1,I))GOTO 2060
         RDATE(7,I)=RDATE(7,I)+(RDATE(1,I)-RDATE(1,1))*365.0
         IF(RDATE(1,1).EQ.(FLOAT(INT(RDATE(1,1)/4.0))*4.0))RDATE(7,I)=
     +   RDATE(7,I)+1.0
2060     GOTO IGO(340,940)
C
C ---------------------------------------------------------------------
C
2100     RETURN
         END
```

```
;
;    TASK BUILD INDIRECT COMMAND FILE FOR NOANAV : NAVIGATION OF NOAA IMAGES
;
NOANAV.TSK/FP/CP=NOANAV.ODL/MP
;
UNITS=6
ASG=SY:1,SY:2,SY:3,SY:4,TT0:5,TT1:6
;
; GET OVERLAY INFORMATION FROM NOANAV.ODL
; END




;
;    OVERLAY DESCRIPTOR FILE FOR NOANAV - NAVIGATION OF NOAA IMAGES.
;
        .ROOT NOANAV-*(N1,N2,N3,N4,N5,N6)
;
N1:     .FCTR NOAN1
N2:     .FCTR NOAN2
N3:     .FCTR NOAN3
N4:     .FCTR NOAN4
N5:     .FCTR NOAN5
N6:     .FCTR NOAN6
;
.END
;
```

APPENDIX B

Listing of program 'GCPFIL'

```
C
C       PROGRAM GCPFIL.FTN
C       THIS PROGRAM SETS UP A FILE WITH GROUND CONTROL POINTS SUITABLE FOR
C       USE WITH TASK GI, TO CALCULATE A POLINOMIAL FOR TRANSFORMING NOAA
C       AVHRR IMAGES TO MERCATOR PROJECTION.
C
C       THE PROGRAM USES TWO OF THE NOANAV SUBROUTINES - AS WAS MODIFIED FOR
C       USE WITH LLGRID - TO COMPUTE THE LATS/LONGS AND LINE/PIXELS. IT IS
C       NECESSARY TO RUN NOANAV , PRIOR TO GCPFIL, IN ORDER TO SET UP THE
C       SATELLITE EPHEMERIS, CALCULATE NODAL LONGITUDE AND TRAVEL TIME AND
C       SAVE ALL THIS IN FILE XXXNAV.DAT ( XXX= AREA CODE FOR THE IMAGE)
C
C       THE USER SPECIFIES LATITUDES ALONG WHICH GCPS ARE TO BE EXTRACTED
C       AND THE NUMBER OF GCPS TO BE EXTRACTED ON EACH LINE OF LATITUDE.
C
C
C       SUBROUTINES USED :
C         1. ARIES II MODULES : ZFSEEK,ZHRDIM,
C         2. SMERCA - CALCULATES DISTANCE OF A LATITUDE FROM EQUATOR IN
C                     MINUTES OF LONGITUDE, FOR A MERCATOR PROJECTION.
C         3. NOANAV SUBROUTINES GRID6A AND GRID6B ( AS MODIFIED FOR LLGRID)
C
C       WRITTEN BY J.J.AGENBAG    S.F.R.I. NOVEMBER 1989 IN FORTRAN 77
C
C
      PROGRAM GCPFIL
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),REAL*8(X-Z)
      DIMENSION BFIL(9),BFOUND(9),XDATW(8),XDATC(10),RCLL(2,4),
     + GCPLAT(20),RARR(3,80)
      CHARACTER CFILI*10,CFILO*13,NAMMOD*7,CARS*3,CARM*3
      DATA BFIL/'*','*','*','F','F','*','*','*','*'/
      CFILI='***NAV.DAT'                  ! INPUT FILE FOR NAVIGATION DATA
      CFILO='***GC***0.GCF'               ! GCP OUTPUT FILE
      LUNIMG=1                            ! LUN FOR IMAGE DISK
      IERR=1
      RPI=4.0*ATAN(1.0)                   ! PI
      DR=RPI/180.0                        ! DEGREES TO RADIAN CONVERSION FACTOR
      REQR=3963.3                 ! EARTH EQ.RADIUS IN MILES
      RPOR=3949.8                 !    ,,  POLAR RADIUS (MILES)
      RECC=SQRT((REQR-RPOR)*(REQR+RPOR))/REQR  ! ECCENTRICITY OF EARTH.
C
C       1.0 ENTER THE 3 CHAR. AREA CODE FOR THE IMAGE TO BE TRANSFORMED.
C           READ FILE DIMENSIONS.
C
      WRITE(5,20)
20    FORMAT(1X//' NB !! TO USE THIS PROGRAM A XXXNAV.DAT FILE',
     +' SHOULD HAVE',/,
     +' BEEN CREATED BY NOANAV. ALSO THE IMAGE SHOULD BE ORIENTATED',/
     +' WITH NORTH AT THE TOP.',/,
     +' DO YOU WISH TO CONTINUE ?  (Y/N):',$)
      READ(5,30)CHAR
30    FORMAT(A1)
      IF(CHAR.NE.'Y')GOTO 2000          ! EXIT
C
40    WRITE(5,50)
50    FORMAT(1X/' ENTER THE 3 CHAR. AREA CODE FOR THE IMAGE TO BE ',
     +'TRANSFORMED :',$)
```

```
      READ(5,60)CARS
60    FORMAT(A3)
      DO 65 I=1,3
        WRITE(CHAR,30)CARS(I:I)
        READ(CHAR,30)BFIL(I)
65    CONTINUE
C
      CALL ZFSEEK(BFIL,1,QQQ,IERR,ISIZE,IBH,IBL,BFOUND) ! SEE IF FILE EXIS
      IF(IERR.EQ.1)THEN  ! FILE EXIST
         CALL ZHRDIM(BFOUND,LUNIMG,QQQ,IERR,NLIN,NPIX,IFP,IFL,ILP,ILL)
         IF(IERR.EQ.1)GOTO 100
         NAMMOD='ZHRDIM '
         GOTO 1000              ! EXIT WITH ERROR.
        ELSE IF(IERR.EQ.-261)THEN     ! FILE DOES NOT EXIST.
         WRITE(5,70)
70       FORMAT(1X//' SPECIFIED FILE NO FOUND.',/,
     +   ' RESPECIFY FILE NAME (R) OR EXIT (E) ? (R/E):',$)
         READ(5,30)CHAR
         IF(CHAR.EQ.'R')GOTO 40
         GOTO 2000              ! EXIT
        ELSE
         NAMMOD='ZFSEEK '
         GOTO 1000              ! EXIT WITH ERROR.
      END IF
C
C     2.0 READ DATA FROM APPROPRIATE NOANAV FILE.
C
100   WRITE(5,110)
110   FORMAT(1X/' ENTER 3 CHAR. AREA CODE FOR THE APPROPRIATE DATA',/,
     + ' FILE CREATED BY NOANAV :',$)
      READ(5,60)CFILI(1:3)
      WRITE(5,114)CFILI
114   FORMAT(1X/' OPEN INPUT DATA FILE ',A10)
      OPEN(4,FILE=CFILI,STATUS='OLD',ERR=120)
      GOTO 160
120   WRITE(5,140)CFILI
140   FORMAT(1X//' UNABLE TO OPEN FILE ',A10,/,
     + ' RESPECIFY NAME (R) OR EXIT (E) ? (R/E):',$)
      READ(5,30)CHAR
      IF(CHAR.EQ.'R')GOTO 100
      IF(CHAR.EQ.'E')GOTO 2000                  ! EXIT
      GOTO 120
160   READ(4,*)XA,XA,XA,XA,XA,XA,XA,XA,XDATW,XDATC
      CLOSE(4)
C
C     3.0 GET AREA CODE FOR THE MASTER IMAGE (YYY)AND CREATE AN OUTPUT FILE
C         WITH NAME XXXGCYYY0.DAT. WRITE HEADER INFORMATION.
C
170   WRITE(5,180)
180   FORMAT(1X/' ENTER 3 CHAR. AREA CODE FOR THE MASTER IMAGE- IE',/,
     + ' THE FILE TO CONTAIN THE TRANSFORMED IMAGE :',$)
      READ(5,60)CFILO(6:8)
      DO 190 I=1,3
        CFILO(I:I)=CARS(I:I)  ! CONSTUCT NAME FOR OUTPUT GCP FILE.
190   CONTINUE
      OPEN(4,FILE=CFILO,STATUS='NEW')
      WRITE(5,200)CFILO
200   FORMAT(1X//' OPENED GCP FILE ',A13)
      WRITE(4,220)CFILO,CFILO(1:3),CFILO(6:8)
220   FORMAT(' GCP FILE ',A13,/,' SLAVE AREA ',A3,/,' MASTER AREA ',
```

```
     +   A3,/,1X,9X,'N=Y/N IMAGE TO UTM FLAG')
C
C        4.0 ENTER THE SCALING FACTOR - IE ONE MINUTE OF LONGITUDE
C            EQUIVALENT TO THIS NUMBER OF PIXELS IN THE OUTPUT IMAGE.
C            COMPUTE THE CORNER LATS/LONGS FOR THE SLAVE IMAGE (INPUT). ROUND
C            TO THE NEXT SMALLER OR LARGER 0.1 DEG. FROM THESE AND THE SCALING
C            FACTOR DETERMINE THE MASTER (OUTPUT) IMAGE DIMENSIONS.
C
C        4.1 SCALING FACTOR.
C
300      WRITE(5,320)
320      FORMAT(1X/' ENTER THE SCALING FACTOR( NUMBER OF PIXELS EQUAL',
     +   ' TO ONE MINUTE',/,' OF LONGITUDE (0.1 TO 10):',$)
         READ(5,*)RSF
         IF(RSF.LT.0.1.OR.RSF.GT.10.0)GOTO 300
C
C        4.2 COMPUTE THE CORNER COORDINATES.
C
         DO 360 I=1,4
           IF(I.EQ.1)THEN          ! 1=TOP LEFT CORNER
             XLIN=DBLE(IFL)
             XPIX=DBLE(IFP)
            ELSE IF(I.EQ.2)THEN    ! 2=TOP RIGHT CORNER
             XLIN=DBLE(IFL)
             XPIX=DBLE(ILP)
            ELSE IF(I.EQ.3)THEN    ! 3=BOTTOM LEFT CORNER
             XLIN=DBLE(ILL)
             XPIX=DBLE(IFP)
            ELSE                   ! 4=BOTTOM RIGHT CORNER
             XLIN=DBLE(ILL)
             XPIX=DBLE(ILP)
           END IF
           CALL GRID6A(XDATW,XDATC,XLIN,XPIX,QQQ,XLAP,XLOP,IERR)
           IF(IERR.NE.1)THEN
             WRITE(5,340)I,XLIN,XPIX
340          FORMAT(1X/' COMPUTATION OF CORNER',I2,' COORDINATES FOR',/,
     +       ' LINE',F8.2,' PIXEL',F8.2,' FAILED.  ABORT.')
             GOTO 1000             ! EXIT WITH ERROR
           END IF
           RCLL(1,I)=XLAP
           RCLL(2,I)=XLOP
360      CONTINUE
C
         RNLAT=MAX(RCLL(1,1),RCLL(1,2),RCLL(1,3),RCLL(1,4)) ! MOST NORTHERLY LA
         RSLAT=MIN(RCLL(1,1),RCLL(1,2),RCLL(1,3),RCLL(1,4)) ! MOST SOUTHERLY LA
         RWLON=MAX(RCLL(2,1),RCLL(2,2),RCLL(2,3),RCLL(2,4))! MOST WESTERLY LONG
         RELON=MIN(RCLL(2,1),RCLL(2,2),RCLL(2,3),RCLL(2,4))! MOST EASTERLY LONG
C
C        4.2.1 FIND LATITUDE FOR MOST WESTERLY CORNER.
C
         DO 370 I=1,4
           IF(RWLON.EQ.RCLL(2,I))THEN
             RLATWC=RCLL(1,I)
             GOTO 375
           END IF
370      CONTINUE
         RLATWC=RSLAT
C
375      WRITE(5,380)(I,(RCLL(J,I),J=1,2),I=1,4),RNLAT,RSLAT,RWLON,RELON,
     +   RLATWC
```

```
380     FORMAT(1X//' CORNER COORDINATES FOR SLAVE IMAGE ARE :',
     +  4(/,' CORNER NO',I2,' LAT =',F9.4,'  LONG =',F9.4),/,
     +  ' NORTHERLY LAT =',F9.4,'  SOUTHERLY LAT =',F9.4,/,
     +  ' WESTERLY LONG =',F9.4,'  EASTERLY LONG =',F9.4,//,
     +  ' LATITUDE FOR MOST WESTERLY CORNER =',F9.4)
C
C       4.3 ROUND THE SOUTHERLY LAT AND EASTERLY LONG UP TO NEXT 0.1 DEG,
C           ROUND THE NORTHERLY LAT AND WESTERLY LONG DOWN TO NEXT 0.1 DEG.
C
        RNLAT=FLOAT(INT(RNLAT*10.0))/10.0        ! FOR S-HEMISPHERE
        RSLAT=FLOAT(INT(RSLAT*10.0))/10.0        ! FOR N-HEMISPHERE
        IF(RNLAT.GE.0.0)RNLAT=RNLAT+0.1
        IF(RSLAT.LT.0.0)RSLAT=RSLAT-0.1
C
        RWLON=FLOAT(INT(RWLON*10.0))/10.0        ! FOR E-HEMISPHERE
        RELON=FLOAT(INT(RELON*10.0))/10.0        ! FOR W-HEMISPHERE
        IF(RWLON.GE.0.0)RWLON=RWLON+0.1
        IF(RELON.LT.0.0)RELON=RELON-0.1
C
C       4.4 CALCULATE THE OUTPUT IMAGE FILE DIMENSIONS FOR A MERCATOR
C           PROJECTION WITH A LATITUDE AND LONGITUDE RANGE AS IN 4.3 AND FOR
C           SCALING FACTOR RSF.
C
        JFP=1              ! FIRST PIXEL
        JLP=INT((ABS(RELON-RWLON)*60.0)*RSF+1.0+0.5)     ! LAST PIXEL
        JFL=1              ! FIRST LINE
        CALL SMERCA(RNLAT,QQQ,RDIST)
        SLIN1=RDIST        ! LINE NUMBER 1 CORRESPONDS TO THIS VALUE
        CALL SMERCA(RSLAT,QQQ,RDIST)
        JLL=INT((ABS(SLIN1-RDIST))*RSF+1.0+0.5) ! LAST LINE
        WRITE(5,400)RNLAT,RSLAT,RWLON,RELON,JFL,JLL,JFP,JLP
400     FORMAT(1X/' ROUNDED LAT/LONG LIMITS FOR OUTPUT IMAGE ARE :',/,
     +  ' NORTH LAT =',F9.4,'  SOUTH LAT =',F9.4,/,
     +  ' WEST LONG =',F9.4,'  EAST LONG =',F9.4,/,
     +  ' LINE',I5,' TO',I5,'  PIXEL',I5,' TO',I5)
C
C       4.5 SPECIFY THE LATITUDES TO BE USED FOR GCP EXTRACTION AND THE
C           NUMBER OF GCP'S PER LATITUDE.
C
405     WRITE(5,410)
410     FORMAT(1X/' NUMBER OF GC-POINTS PER LINE OF LATITUDE ?',
     +  ' (MAX=10):',$)
        READ(5,*)NUMPT
        IF(NUMPT.LT.2.OR.NUMPT.GT.10)GOTO 405
420     WRITE(5,440)NUMPT
440     FORMAT(1X//' NUMBER OF LATITUDE LINES ALONG WHICH GCP POINTS ',
     +  'ARE TO BE COMPUTED(MAX=20)',/,
     +  ' (',I3,' GCPS WILL BE COMPUTED PER LATITUDE ) :',$)
        READ(5,*)NUMLAT
        IF(NUMLAT.LT.1)THEN
          WRITE(5,460)
460       FORMAT(' DO YOU WISH TO EXIT ? (Y/N) :',$)
          READ(5,30)CHAR
          IF(CHAR.EQ.'Y')GOTO 2000          ! EXIT
          GOTO 420
         ELSE IF(NUMLAT.GT.20)THEN
          GOTO 420
        END IF
C
        DO 540 I=1,NUMLAT
```

```
500       WRITE(5,520)I
520       FORMAT(1X/' LATITUDE NO.',I3,' :  ENTER DECIMAL DEGREE :',$)
          READ(5,*)GCPLAT(I)
          IF(GCPLAT(I).LE.RNLAT.AND.GCPLAT(I).GE.RSLAT)GOTO 540
          WRITE(5,530)RNLAT,RSLAT
530       FORMAT(1X//' LATITUDE OUT OF RANGE. THE RANGE IS',F9.4,' TO',
     +    F9.4)
          GOTO 500
540     CONTINUE
C
C       4.6 CALCULATE THE LINE AND PIXEL NUMBERS, IN THE MASTER IMAGE,
C       CORRESPONDING TO FULL DEGREE LATS AND LONGS. LIST THESE ON THE PRINTE
C       FOR ANNOTATION OF THE TRANSFORMED - IE MASTER - IMAGE.
C
C       4.6.1 LATITUDES/LINES
C
        WRITE(5,544)
544     FORMAT(1X//' PRODUCE  A LIST OF LATS VS LINE NUMBERS AND LONGS'/
     +  ' VS PIXEL NUMBERS FOR THE MASTER IMAGE ? (Y/N) :',$)
        READ(5,30)CHAR
        IF(CHAR.EQ.'N')GOTO 610
C
        WRITE(6,550)CFILO(6:8)
550     FORMAT(1X//' LINE AND PIXEL NUMBERS CORRESPONDING TO LATS AND ',
     +  'LONGS IN IMAGE ',A3,/,1X,68('*'),/,
     +  1X,10X,'LATITUDE (DEGREES)',5X,'LINE NUMBER',/,
     +  1X,10X,18('-'),5X,11('-'),/)
        I1=INT(RNLAT)              ! ROUND NORTHERN LAT TO FULL DEGREE
        IF(RNLAT.LT.0.0)I1=INT(RNLAT-0.9)
        I2=INT(RSLAT)              ! ROUND SOUTHERN LAT TO FULL DEGREE.
        IF(RSLAT.GT.0.0)I2=INT(RSLAT+0.9)
        DO 570 I=I1,I2,-1
          CALL SMERCA(FLOAT(I),QQQ,RDIST)
          RMLIN=ABS(SLIN1-RDIST)*RSF+1.0
          WRITE(6,560)FLOAT(I),RMLIN
560       FORMAT(1X,10X,F11.0,12X,F8.1)
570     CONTINUE
C
C       4.6.2 LONGITUDES/PIXELS
C
        WRITE(6,580)
580     FORMAT(1X/1X,9X,'LONGITUDE (DEGREES)',4X,'PIXEL NUMBER',/,
     +  1X,9X,19('-'),4X,12('-'),/)
        I1=INT(RWLON)              ! ROUND WESTERN LONG TO FULL DEGREE.
        IF(RWLON.LT.0.0)I1=INT(RWLON-0.9)
        I2=INT(RELON)              ! ROUND EASTERN LONG TO FULL DEGREE.
        IF(RELON.GT.0.0)I2=INT(RELON+0.9)
        DO 600 I=I1,I2,-1
          RMPIX=ABS((FLOAT(I)-RWLON)*60.0)*RSF+1.0
          WRITE(6,560)FLOAT(I),RMPIX
600     CONTINUE
610     CONTINUE
C
C       5.0 EXTRACT GC-POINTS ALONG THE SPECIFIED LATITUDES. IN EACH CASE
C       FIND A LONGITUDE - TO THE NEAREST 0.1 DEG - WHICH CORRESPONDS TO A
C       PIXEL INSIDE THE SPECIFIED INPUT FILE(FROM SUB GRID6B).
C       THIS GIVES THE SLAVE LINE AND PIXEL.  PROCEED THEN IN STEPS OF HALF
C       DEGREES OF LONGITUDE UNTIL THE EASTERN EDGE IS EXCEEDED. THEN FIND ONE
C       POINT (TO THE NEAREST 0.1 DEG) JUST INSIDE THE IMAGE.
C       STORE THESE IN RARR(J,I) WHERE J: 1=LONGITUDE, 2=LINE, 3=PIXEL;
```

```
C       I=SUCESSIVE CONTROL POINTS. ON COMPLETION OF THE LINE, SELECT THE
C       FIRST AND LAST POINTS AS WELL AS NUMPT-2 OTHER POINTS, EVENLY SPREAD
C       ( IN TERMS OF GCP NUMBER) BETWEEN NUMBER 1 AND THE LAST. COMPUTE THE
C       MASTER IMAGE PIXEL NUMBER FOR EACH AND TRANSFER TO THE GCP FILE.
C
        NGCP=0              ! COUNT NUMBER OF GCPS EXTRACTED
        RSLON=RWLON
        DO 900 I=1,NUMLAT
          XLAT=GCPLAT(I)
          RLAT=GCPLAT(I)
          WRITE(5,640)XLAT
C         WRITE(6,640)XLAT
640       FORMAT(1X/' EXTRACT GCPS ON LATITUDE',F6.2,' :')
          NGCPL=0
          CALL SMERCA(RLAT,QQQ,RDIST)
          RMLIN=ABS(SLIN1-RDIST)*RSF+1.0  ! MASTER IMAGE LINE NO. FOR THIS LA
C         WRITE(6,650)RLAT,RDIST,RMLIN
C650      FORMAT(1X/' 650 : RLAT=',F6.2,'  RDIST=',F10.4,'  RMLIN=',
C    +    F8.2)
C
C       5.1 FIND FIRST LONGITUDE JUST INSIDE THE IMAGE. FOR THE FIRST LATITUD
C       USE THE MOST WESTERLY POINT (RWLON) TO START THE SEARCH. FOR SUBSEQUE
C       LATITUDES USE THE START POINT FROM THE PREVIOUS LINE. HOWEVER, IF NOR
C       OF THE MOST WESTERLY CORNER ( LAT=RLATWC ) CONTINUE USING RWLON TO
C       START THE SEARCH.
C
          XINT=1.0D0
          XLON=RSLON
C         WRITE(6,670)RLAT,RLATWC,XLON,RWLON
C670      FORMAT(1X/' 670:  RLAT,RLATWC,XLON,RWLON=',4F10.4)
          RPL=MAX(IFP-100,1)
          YPL=DBLE(RPL)
700       YPH=YPL+300.0D0
710       CONTINUE
C         WRITE(6,715)IFP,IFL,RPL,YPL,YPH
C715      FORMAT(' 715 : IFP,IFL=',2I6,'  RPL,YPL,YPH=',3F8.2)
          CALL GRID6B(XDATW,XDATC,-XLAT,-XLON,YPL,YPH,QQQ,ILIN,IPIX)
C         WRITE(6,720)XINT,YPL,YPH,XLAT,XLON,ILIN,IPIX,IFL,IFP
C720      FORMAT(1X/' 720: XINT=',F5.2,' YPL,YPH=',2F8.2,/,
C    +    ' XLAT=',F8.4,' XLON=',F8.4,' ILIN,IPIX=',2I6,/,
C    +    ' IFL,IFP=',2I6)
          IF(ILIN.EQ.0.AND.IPIX.EQ.0)THEN
            YPL=0.0D0
            YPH=2049.0
            GOTO 710
          END IF
          IF(ILIN.GE.IFL.AND.ILIN.LE.ILL.AND.IPIX.GE.IFP.AND.IPIX.LE.
     +    ILP)THEN
             IF(XINT.EQ.0.1D0)GOTO 800
             IF(XINT.EQ.0.5D0)XINT=0.1D0
             IF(XINT.EQ.1.0D0)XINT=0.5D0
             YPL=YPLO
             XLON=XLONO-XINT
            ELSE
             YPLO=  YPL
             YPL=DBLE(MAX(IPIX-100,1))
             XLONO=  XLON
             XLON=  XLON-XINT
          END IF
          GOTO 700
```

```
C
C       5.2 FIRST LONGITUDE FOUND. STORE IN RARR().
C
800       RSLON=XLON
          IF(RLAT.GE.RLATWC)RSLON=RWLON
          NGCPL=NGCPL+1          ! NUMBER OF GCPS FOR THIS LATITUDE.
          RARR(1,1)=SNGL(XLON)
          RARR(2,1)=FLOAT(ILIN)
          RARR(3,1)=FLOAT(IPIX)
C         WRITE(6,820)XLAT,(RARR(J,1),J=1,3)
C820      FORMAT(1X/' 820: LAT=',F8.4,'  1ST LONG. : RARR(J,1)=',F7.2,
C    +    2F7.1)
C
C       5.3 DO REST OF THE LAT. LINE IN STEPS OF HALF DEGREES LONGITUDE UNTIL
C       THE EASTERN EDGE IS REACHED.
C
840       XLONT=XLON
          XLON=DBLE(INT(XLON-0.05D0))
          IF(ABS(XLON-XLONT).GE.0.5D0)XLON=XLON-0.5D0
          XLON=XLON-0.5D0
850       YPL=DBLE(IPIX)
          YPH=YPL+200.0D0
852       CALL GRID6B(XDATW,XDATC,-XLAT,-XLON,YPL,YPH,QQQ,ILIN,IPIX)
C         WRITE(6,854)XLAT,XLON,YPL,YPH,ILIN,IPIX
C854      FORMAT(1X/' 854: XLAT,XLON,YPL,YPH,ILIN,IPIX=',2F7.2,
C    +    2F7.1,2I6)
          IF(ILIN.EQ.0.AND.IPIX.EQ.0)THEN
            YPL=0.0D0
            YPH=2049.0D0
            GOTO 852
          END IF
          IF(ILIN.GE.IFL.AND.ILIN.LE.ILL.AND.IPIX.GE.IFP.AND.
     +    IPIX.LE.ILP)THEN              ! INSIDE IMAGE AREA
             NGCPL=NGCPL+1
             RARR(1,NGCPL)=SNGL(XLON)
             RARR(2,NGCPL)=FLOAT(ILIN)
             RARR(3,NGCPL)=FLOAT(IPIX)
C            WRITE(6,856)XLAT,NGCPL,(RARR(J,NGCPL),J=1,3)
C856         FORMAT(1X/' 856: LAT=',F8.4,'  LONG NO.',I4,' RARR=',
C    +       F7.2,2F7.1)
             XLON=XLON-0.5D0
             GOTO 850
            ELSE                    ! OFF EASTERN EDGE. FIND LONGITUDE NEAR EDGE.
860          XLON=XLON+0.1D0
             CALL GRID6E(XDATW,XDATC,-XLAT,-XLON,YPL,YPH,QQQ,ILIN,IPIX)
C            WRITE(6,864)XLAT,XLON,YPL,YPH,ILIN,IPIX
C860         FORMAT(1X/' 864: LAT=',F8.4,'  XLON,YPL,YPH,ILIN,IPIX=',
C    +       F7.2,2F7.1,2I6)
             IF(ILIN.GT.ILL.OR.IPIX.GT.ILP)GOTO 860
             NGCPL=NGCPL+1
             RARR(1,NGCPL)=SNGL(XLON)
             RARR(2,NGCPL)=FLOAT(ILIN)
             RARR(3,NGCPL)=FLOAT(IPIX)
          END IF
C
C         5.4 SELECT NUMPT POINTS AND TRANSFER TO THE GCP FILE.
C
C         WRITE(6,866)NGCPL,NUMPT,((RARR(J,K),J=1,3),K=1,NGCPL)
C866      FORMAT(1X/' 866: NGCPL,NUMPT=',2I3,'  RARR= :',
C    +    (/,1X,3(F6.2,2F8.1,4X)))
```

```
          IF(NGCPL.LE.NUMPT)THEN          ! LESS OR EQUAL THE SPECIFIED
             DO 868 J=1,NGCPL             ! NUMBER OF POINTS EXTRACTED.
               NGCP=NGCP+1                ! USE ALL AVAILABLE POINTS.
               RMPIX=ABS((RARR(1,J)-RWLON)*60.0)*RSF+1.0 ! MASTER PIXEL
               WRITE(4,870)RARR(3,J),RARR(2,J),RMPIX,RMLIN,NGCP
868          CONTINUE
            ELSE                          ! EXTRACT REQUIRED NUMBER BY
             RINC=FLOAT(NGCPL-1)/(FLOAT(NUMPT-1))     ! INTERPOLATION OF
             RP=1.0-RINC                              ! GCP NUMBERS.
             DO 880 J=1,NUMPT
               RP=RP+RINC
               IP=INT(RP)
               NGCP=NGCP+1
               RMPIX=ABS((RARR(1,IP)-RWLON)*60.0)*RSF+1.0 ! MASTER PIXEL
               WRITE(4,870)RARR(3,IP),RARR(2,IP),RMPIX,RMLIN,NGCP
870            FORMAT(4F17.6,I6)
880          CONTINUE
          END IF
C
C       END OF THIS LATITUDE. GO DO THE NEXT
C
900     CONTINUE
C
C       6.0 END OF GCP EXTRACTION. CLOSE FILE. EXIT
C
        CLOSE(4)
        WRITE(5,920)NGCP
920     FORMAT(1X//' END OF GCP EXTRACTION. TOTAL NUMBER OF GCPS=',I6)
        GOTO 2000
C
C       6.1 EXIT WITH ERROR
C
1000    WRITE(5,1020)IERR,NAMMOD
1020    FORMAT(1X//' EXIT WITH ERROR NUMBER',I5,' IN MODULE ',A7)
C
C       6.2 NORMAL EXIT
C
2000    CALL EXIT
        END
```

```
C
C       SUBROUTINE SMERCA.FTN
C
C       SUBROUTINE USED BY PROGRAM GCPFIL TO CALCULATE DISTANCES OF
C       LATITUDE LINES FROM THE EQUATOR ON A MERCATOR PROJECTION IN MINUTES
C       OF LONGITUDE.
C
C       J.J.AGENBAG  SFRI NOVEMBER 1989
C
        SUBROUTINE SMERCA(RLAT,QQQ,RDIST)
C
C       RLAT IN DECIMAL DEGREES
C
        RPI=4.0*ATAN(1.0)          !  PI
        DR=RPI/180.0               ! DEGREES TO RADIANS CONVERSION.
        REQR=3963.3       ! EARTH EQUATORIAL RADIUS.
        RPOR=3949.8       ! POLAR RADIUS
        RECC=SQRT((REQR-RPOR)*(REQR+RPOR))/REQR ! ECCENTRICITY OF EARTH
        RSCALE=(2.0*RPI*REQR)/(360.0*60.0)
        RRLAT=RLAT*DR              ! LATITUDE IN RADIANS
        RTAN=TAN((RPI/4.0)+(RRLAT/2.0))
        RSIN=SIN(RRLAT)
        RFAC=((1.0-RECC*RSIN)/(1.0+RECC*RSIN))**(RECC/2.0)
        RDIST=REQR*LOG(RTAN*RFAC)/RSCALE
        RETURN
        END
```

```
C
C       SUBROUTINE GRID6A.FTN
C       A SUBPROGRAM TO LLGRID. THIS SUBROUTINE CALCULATES THE LAT. AND
C       LONG. FOR A GIVEN LINE/PIXEL COORDINATE.
C
C       J.J.AGENBAG   S.F.R.I     AUGUST 1989.     IN FORTRAN 77
C       ( THIS IS A MODIFICATION OF SUBROUTINE NOAN2  CREATED JUNE 1989)
C
        SUBROUTINE GRID6A(XDATW,XDATC,XLIN,XPIX,QQQ,XLAP,XLOP,IERR)
C
        IMPLICIT REAL*8(X-Z),CHARACTER*1(C)
        DIMENSION XDATW(8),XDATC(10)
C
        IERR=1
        XDR=XDATC(8)
C       WRITE(6,5)XDATW(1),XDATW(2)/60.0D0,XDATW(3)/XDR,XDATW(4),
C     +  XDATW(5)/XDR,XDATW(6),XDATW(7)/XDR,XDATC(1)/XDR,XDATC(2),
C     +  XDATC(3)/XDR,XDATC(4)/XDR,XDATC(5)/XDR,(XDATC(I),I=6,10)
C5      FORMAT(' 1. XDATW =:',/,' 1=',F12.6,'  2=',F12.6,'  3=',F12.6,/,
C     +  ' 4=',F12.8,'  5=',F12.6,'  6=',F10.8,'  7=',F12.8,/,
C     +  ' XDATC =:',/,' 1=',F12.6,' 2=',F13.6,'  3=',F12.6,/,
C     +  ' 4=',F12.6,'  5=',F12.6,'  6=',F12.6,'  7=',F15.8,/,
C     +  ' 8=',F15.8,'  9=',F15.8,'  10=',F15.8)
C
C
C       CALCULATE SOME CONSTANTS
C
        XRN=DSIGN(1.0D0,XDATW(3))       ! 1= ASCENDING, -1= DESCENDING TRACK
        XC1=(1.0D0-XDATW(4)**2.0)*XDATW(1)
        XOPI=2.0D0*XDATC(7)
        XINC=(XDATW(3)-XRN*XDATC(7)/2.0D0) ! INCLINATION W OF N (RAD).
C
C
C ------------------ START CALCULATION CYCLE -------------------------
C
C
C
C       1.0 FIND TRUE ANOMALY PERIFOCUS TO GP .
C
        XT=XDATC(2)-XRN*(XLIN-1.0D0)*XDATW(6) ! TRAVEL TIME EQUATOR TO GP
        XT1=XT+XDATC(6)          ! TRAVEL TIME PERIFOCUS TO GP
        IF(XT1.LT.0.0D0)XT1=XT1+XDATW(2)
        IF(XT1.GE.XDATW(2))XT1=XT1-XDATW(2)
        ZTF=XDATC(3)*XT1
C       FIND GP ECCENTRIC ANOMALY,XE1, BY ITERATION FROM THE EQUATION
C       XE1-E*SIN(XE1)=ZTF     E=ECCENTRICITY
C
        IRD=1             ! ITERATION STEP COUNTER
        XE1=ZTF           ! FIRST APPROXIMATION
240     ZE1=XE1-(XE1-XDATW(4)*DSIN(XE1)-ZTF)/(1.0D0-XDATW(4)*DCOS(XE1))
        IF(DABS(XE1-ZE1).LE.0.1D-5)GOTO 280 ! SUCCESS. END ITERATION
        IRD=IRD+1
        XE1=ZE1
        IF(IRD.LE.20)GOTO 240    ! NEXT ITERATION
        WRITE(5,260)
260     FORMAT(1X/' ERROR !!. ITERATION FOR ECCENTRIC ANOMALY FAILED.')
        IERR=1000
        GOTO 600
C
```

```
280      CONTINUE
C        WRITE(6,285)NUM,XLIN,XPIX,XT,ZTF/XDR,IRD,XE1/XDR
C285     FORMAT(1X//' 2. NUM=',I4,' XLIN=',F12.6,'  XPIX=',F12.6,/,
C     +  ' XT=',F16.8,'  ZTF=',F15.8,'  IRD=',I3,/,
C     +  ' XE1=',F15.8)
         XCV=(DCOS(XE1)-XDATW(4))/(1.0D0-XDATW(4)*DCOS(XE1)) ! COS TRUE ANOMA
         XV=DACOS(XCV)    ! TRUE ANOMALY FOR GP.
         IF(XE1.GT.XDATC(7))THEN
           XV=X2PI-XV
           XCV=DCOS(XV)
         END IF
C
C        3.0 CALCULATE GP. LAT AND LONG.
C
         IF(XRN.GE.0.0D0)THEN     ! ASCENDING TRACK
            XTETA=XV-X2PI+XDATW(5) !GEOCENTRIC ANGLE EQUATOR TO GP.
            IF(XTETA.LT.-XDATC(7))XTETA=XTETA+X2PI
           ELSE                   ! DESCENDING TRACK.
            XTETA=XV-XDATW(7)+XDATW(5)
         END IF
         IF(XTETA.GE.XDATC(7))XTETA=XTETA-X2PI
         XLAT=DASIN(DCOS(XINC)*DSIN(XTETA))  ! GP. LATITUDE
         XINCO=DASIN(DSIN(XINC)/DCOS(XLAT))  ! TRACKLINE INCLINATION AT GP
         XDLAM=DASIN(DTAN(XINC)*DTAN(XLAT)) !LONG. DIFF. EQUAT. TO GP
         XLON=XDATC(1)+XDLAM+XT*XDATC(4)  ! GP. LONGITUDE
         XRD=XC1/(1.0D0+XDATW(4)*XCV) ! RADIAL DISTANCE, EARTH CENTRE TO SAT.
C
C        WRITE(6,290)XV/XDR,XTETA/XDR,XLAT/XDR,XINCO/XDR,XLAM/XDR,
C     +  XLON/XDR,XRD
C290     FORMAT(1X/' 3.  XV=',F12.6,'  XTETA=',F12.6,'  XLAT=',F12.6,/,
C     +  ' XINCO=',F12.6,'  XDLAM=',F15.8,'  XLON=',F12.6,'  XRD=',F12.6)
C
C        4.0 CALCULATE PIXEL LAT AND LONG. THE CALCULATION INVOLVES THE
C            EARTH RADIUS AT THE (UNKNOWN) PIXEL LATITUDE. FIRST CARRY
C            THROUGH THE CALCULATION USING THE RADIUS AT THE GP. THEN REPEAT
C            WITH THE DERIVED PIXEL LATITUDE.
C
         XNU=(XPIX-1024.5D0)*XDATW(7)     ! SCAN ANGLE : NADIR TO PIXEL
         XTHS=XNU
         XPSI=XDATC(5)    ! AZIMUTH WITH SCAN SCEW =CA. 90.01 DEG (RAD)
         XPSIP=XPSI-XINCO         ! TRUE BEARING , GP TO PIXEL
         IRD=1            ! FIRST CALCULATION
         XLA=XLAT         ! SET PIXEL LAT=GP LAT.
300      XRAD=6378.137D0*(1.0D0-.00335D0*(DSIN(XLA))**2.0) ! EARTH RADIUS(KM.)
         XALTC=XRD/XRAD
         XTH=DASIN(XALTC*DSIN(XTHS))-XTHS  ! GEOCENTRIC ANGLE : GP TO PIXEL
C        WRITE(6,310)IRD,XNU/XDR,XPSI/XDR,XPSIP/XDR,XLA/XDR,
C     +  XRAD,XALTC,XTH/XDR
C310     FORMAT(1X/' 4. IRD=',I3,'  XNU=',F12.6,'  XPSI=',F12.6,/,
C     +  ' XPSIP=',F12.6,'  XLA=',F12.6,/,
C     +  ' XRAD=',F15.6,'  XALTC=',F12.8,'  XTH=',F12.6)
         IF(IRD.GT.2)GOTO 340
         XLAP=DASIN(DCOS(XTH)*DSIN(XLAT)+DSIN(XTH)*DCOS(XLAT)*
     +   DCOS(XPSIP))     ! PIXEL LATITUDE
         XLA=XLAP
         IRD=IRD+1
         GOTO 300         ! REPEAT PIXEL LAT. CALCULATION WITH THIS LAT.
C
340      XDLAM=DASIN(DSIN(XPSIP)*DSIN(XTH)/DCOS(XLAP)) ! LONGITUDE DIFFERENCE
                          ! BETWEEN GP. AND PIXEL.
```

```
      XLOP=(XLON-XDLAM)/XDATC(8)       ! PIXEL LONG. (DEG)
      XLAP=DATAN(DTAN(XLAP)/0.9933D0) ! CONVERT PIXEL LAT. FROM GEOCENTRIC
                                       ! TO GEODETIC.
      XLAP=XLAP/XDATC(8)               ! PIXEL LAT.  (DEG)
C     WRITE(6,350)XLAP,XDLAM/XDR,XLOP
C350  FORMAT(1X/' 5. XLAP=',F12.6,'  XDLAM=',F12.6,' XLOP=',F12.6,/)
C
C
C ----------------------- END CALCULATION CYCLE ----------------------
C
600   RETURN
      END
```

```
C
C       SUBROUTINE GRID6B.FTN
C       A SUBPROGRAM TO LLGRID. THIS ROUTINE CALCULATES THE LINE AND PIXEL
C       NUMBER FOR A GIVEN LAT./LONG. POSITION.
C
C       J.J.AGENBAG     S.F.R.I     AUGUST 1989.    IN FORTRAN 77
C       ( THIS IS A MODIFICATION OF SUBROUTINE NOAN3  CREATED JUNE 1989.)
C
      SUBROUTINE GRID6B(XDATW,XDATC,XLAT,XLON,YPL,YPH,QQQ,ILIN,IPIX)
C
        IMPLICIT REAL*8(X-Z),CHARACTER*1(C)
        CHARACTER DATAFN*10
        DIMENSION XDATW(8),XDATC(10)
C
C
        XDR=XDATC(8)
C       WRITE(6,5)XDATW(1),XDATW(2)/60.0D0,XDATW(3)/XDR,XDATW(4),
C     +  XDATW(5)/XDR,XDATW(6),XDATW(7)/XDR,XDATC(1)/XDR,XDATC(2),
C     +  XDATC(3)/XDR,XDATC(4)/XDR,XDATC(5)/XDR,(XDATC(I),I=6,10)
C5      FORMAT(' 1. XDATW =:',/,' 1=',F12.6,'  2=',F12.6,'  3=',F12.6,/,
C     +  ' 4=',F12.8,'  5=',F12.6,'  6=',F10.8,'  7=',F12.8,/,
C     +  ' XDATC =:',/,' 1=',F12.6,' 2=',F13.6,'  3=',F12.6,/,
C     +  ' 4=',F12.6,'  5=',F12.6,'  6=',F12.6,'  7=',F15.8,/,
C     +  ' 8=',F15.8,'  9=',F15.8,'  10=',F15.8)
C
        XRN=DSIGN(1.0D0,XDATW(3))          ! 1=ASCENDING AND -1=DESCENDING TRAC
        XC1=(1.0D0-XDATW(4)**2.0)*XDATW(1)
        XINC=(XDATW(3)-XRN*XDATC(7)/2.0D0) ! INCLINATION W OF N (RAD).
        X2PI=2.0D0*XDATC(7)
        X90=0.5D0*XDATC(7)                 ! PI/2 RADIANS
C
C
C ----------------- START CALCULATION CYCLE -------------------------
C
C
        XLAP=-XLAT*XDR
        XLAP=DATAN(0.9933D0*DTAN(XLAP)) ! CONVERT PIXEL LAT. FROM GEODETIC
                                        ! TO GEOCENTRIC.
        XLOP=-XLON*XDR
        XRAD=6378.137D0*(1.0D0-.00335D0*(DSIN(XLAP))**2.0) ! EARTH RADIUS
                                        ! AT GIVEN LATITUDE (KM)
C
        ITER=1
        XPL=YPL          ! SET INITIAL PIXEL VALUE
        XPH=YPH                  ! FOR ITERATION
        XPIX=MAX(XPL,XPH)
        XPL=MIN(XPL,XPH)
        XPH=XPIX
500     XPIX=(XPL+XPH)*0.5D0
C
C       CALCULATE GROUND POINT LATITUDE FIRST WITH SATELLITE RADIAL
C       DISTANCE (AND XALTC) FROM PIXEL LATITUDE. THEN RECALCULATE
C       TWICE WITH DERIVED GP. LATITUDE.
C
        IRD=1.           ! FIRST CALCULATION
        XNU=(XPIX-1024.5D0)*XDATW(7)    ! SCAN ANGLE TO PIXEL (RAD)
        XTHS=XNU
        XPSI=XDATC(5)    ! RELATIVE BEARING TO PIXEL =CA 90.01DEG IN RAD
        XRAD=6378.137D0*(1.0D0-.00335D0*(DSIN(XLAP))**2.0) ! EARTH RADIUS
```

```
                                                        ! AT PIXEL
      XLA=XLAP          ! INITIALISE GP. LAT TO PIXEL LAT
      XLAO=XLA          ! SAV PREVIOUS RESULT FOR ITERATION CHECK.
540   XTETA=XRN*DASIN(DSIN(XLA)/DCOS(XINC))  ! ANGULAR DISTANCE EQ. TO XLA
      IF(XRN.GE.0.0D0)THEN              ! ASCENDING TRACK
         XV=X2PI+XTETA-XDATW(5) ! TRUE ANOMALY=ANG. PERIFOCUS TO XLA
         IF(XV.GE.X2PI)XV=XV-X2PI
       ELSE                     ! DESCENDING TRACK
        XV=XDATC(7)+XTETA-XDATW(5)
      END IF
      IF(XV.LT.0.0D0)XV=X2PI+XV
      XCV=DCOS(XV)
      XINCO=DASIN(DSIN(XINC)/DCOS(XLA))
      XPSIP=XPSI-XINCO
      XRD=XC1/(1.0D0+XDATW(4)*XCV)  ! GEOCENTRIC DIST. TO SATELLITE
                                    ! AT THIS LATITUDE.
      XALTC=XRD/XRAD            ! ALTITUDE FACTOR
      XTH=DASIN(XALTC*DSIN(XTHS))-XTHS  ! GEOCENTRIC ANGLE - GP TO PIXEL
      XDLAM=DASIN(DSIN(XPSIP)*DSIN(XTH)/DCOS(XLAP))   ! DELTA LONGITUDE
      XA=(X90-XLAP+XTH)*0.5D0
      XB=(XPSIP-XDLAM)*0.5D0
      XC=(XPSIP+XDLAM)*0.5D0
      XLA=X90-2.0D0*DATAN(DCOS(XC)*DTAN(XA)/DCOS(XB))
      IF(DABS(XLA-XLAO).GT.0.00004D0)THEN
        XLAO=XLA
        IRD=IRD+1
        GOTO 540
      END IF
C
C     CALCULATE GP. LONGITUDE
C
      XLO=XLOP+XDLAM            ! GP. LONGITUDE (RAD)
C
C     CALCULATE EQUATOR CROSSING LONG. AND TRAVEL TIME.
C
      XDLAM=DASIN(DTAN(XLA)*DTAN(XINC)) ! DELTA LONG - GP TO EQ. CROSSING
      XE1=DACOS(XDATW(4)+XRD*XCV/XDATW(1))  ! ECCENTRIC ANOMALY
      IF(XV.GT.XDATC(7))XE1=X2PI-XE1
      XT=XDATC(9)*(XE1-XDATW(4)*DSIN(XE1))-XDATC(6) ! TRAVEL TIME - EQUATOR
                                                    ! TO GP
      IF(DSIGN(1.0D0,XTETA).NE.DSIGN(1.0D0,XT))XT=XT+
     +  DSIGN(1.0D0,XTETA)*XDATW(2)
      XLONE=XLO-XDLAM-XT*XDATC(4) ! EQ. CROSSING LONGITUDE(RAD).
C
C     WRITE(6,580)ITER,XPIX,XLA/XDR,XLO/XDR,XT,XLONE/XDR,XDATC(1)/XDR
C580  FORMAT(' ITERATION NO',I3,' PIXEL=',F10.4,' GP LAT AND LONG=',
C    +  2F10.4,/,' XT=',F12.5,'  XLONE=',F10.5,'  TRUE NOD LONG=',F10.5)
C
C     CHECK AGAINST TRUE NODAL LONGITUDE.
C
      XDIF=XLONE-XDATC(1)
      IF(DABS(XDIF).LT.1.74D-5)GOTO 600       ! 1.74D-5 RAD = 0.01DEG
      ITER=ITER+1
      IF(ITER.GT.20)THEN
          XLIN=0.0D0
          XPIX=0.0D0
          GOTO 800
        ELSE
          IF(XDIF.GT.0.0D0)XPH=XPIX
          IF(XDIF.LT.0.0D0)XPL=XPIX
```

```
            GOTO 500
         END IF
C
C600     XPIX=1024.5D0+XNU/XDATW(7)       ! THE PIXEL NUMBER
600      XLIN=1.0D0+XRN*(XDATC(2)-XT)/XDATW(6)    ! THE LINE NUMBER
800      ILIN=INT(XLIN+0.5D0)
         IPIX=INT(XPIX+0.5D0)
C        WRITE(5,820)XLAT,XLON,XLIN,XPIX,ILIN,IPIX
C820     FORMAT(1X/' GRID6B: XLAT,XLON=',2F10.4,/,
C     +  ' XLIN,XPIX=',2F9.3,'  ILIN,IPIX=',2I6,/)
         RETURN
         END
```

```
;
; TASK BUILD INDIRECT COMMAND FILE FOR GCPFIL - CREATES A FILE WITH GROUND
; CONTROL POINTS FOR COMPUTATION OF A TRANSFORM POLINOMIAL.
;
GCPFIL.TSK/FP/CP=GCPFIL.ODL/MP
;
UNITS=6
ASG=IM:1,DR1:2,DR1:3,DR1:4,TI:5,TT1:6
;
; GET REMAINING INSTRUCTIONS FROM THE OVERLAY DESCRIPTOR FILE,GCPFIL.ODL
;
```

```
;
; OVERLAY DESCRIPTOR FILE FOR GCPFIL - CREATES A FILE WITH GROUND CONTROL
; POINTS USED FOR CALCULATION OF A TRANSFORM POLINOMIAL BY TASK GI.
;
        .ROOT GCPFIL-D-M-*(A,B)
;
D:      .FCTR DR0:[100,300]COMMON/LB:ZFSEEK:ZHRDIM-LIB
M:      .FCTR SMERCA
A:      .FCTR GRID6A
B:      .FCTR GRID6B
;
LIB:    .FCTR DR0:[100,300]NONRLS/LB-DR0:[100,300]COMMON/LB
;
        .END
```

APPENDIX C

Listing of program 'ADVECT'

```
C
C       PROGRAM ADVECT.FTN
C       THIS PROGRAM PERFORMS AUTOMATIC FEATURE TRACKING USING A PROCEDURE
C       DESCRIBED BY EMERY ET. AL. 1986 J.G.R P12865-12878 VOL 91 NO C11.
C       THE PROCEDURE REQUIRES TWO IMAGES SAY 12 HOURS APART, AND EITHER
C       REGISTERED TO EACH OTHER OR TO A COMMON MAP PROJECTION.
C       THE IMAGES SHOULD BE GRADIENT IMAGES AND THE LAND AND CLOUD MASKED
C       BY PIXEL COUNTS OF 255.
C       THE USER THEN SPECIFY THE FILE NAMES AND SECTION OF THE IMAGE TO
C       BE PROCESSED. ALSO THE TEMPLATE WINDOW SIZE N1 (N1 LINES X N1 PIXELS
C       IN THE FIRST IMAGE) AND A SEARCH WINDOW SIZE N2 ( N2 X N2 IN 2'ND
C       IMAGE ). N1<N2 . N1 AND N2 MUST BE EVEN  AND NOT BIGGER THAN 64.
C       N2 MUST ALSO COMPLY WITH THE REQUIREMENTS FOR A FAST FOURIER
C       TRANSFORM IE. IT MUST BE A POWER OF 2 - 8,16,32 OR 64.
C       THE PROGRAM WILL ATTEMPT TO DERIVE THE START AND END POINTS FOR AN
C       ADVECTION VECTOR IN THE WINDOWS ( STARTING AT THE TOP LEFT CORNER
C       OF THE SPECIFIED IMAGE SECTION) USING THE PRINCIPLE OF MAXIMUM
C       CROSS CORRELATION (MCC). IF THE MCC IS LESS THAN 0.4 OR THE MAXIMUM
C       NOT UNIQUE OR IF THE SPECIFIED WINDOWS CONTAIN MORE THAN 1% CLOUD OR
C       LAND THEN NO VECTOR WILL BE CALCULATED. OTHERWISE THE START LINE/PIXE
C       AND END LINE/PIXEL NUMBERS ARE RECORDED IN A FILE XXXVEC.DAT , WHERE
C       XXX IS THE AREA CODE FOR THE SPECIFIED IMAGES. WRITING TO THE FILE
C       IS DONE BY WRITE(4,*)L1,P1,L2,P2
C
C       THE USER ALSO SPECIFY A WINDOW SHIFT, WHICH IS THE NUMBER OF LINES/
C       PIXELS BY WHICH THE WINDOWS ARE SHIFTED ALONG FOR SUCCESSIVE VECTOR
C       DETERMINATIONS - IE. IT DETERMINES POTENTIAL VECTOR DENSITY.
C
C       A TOTAL OF 5 SUBROUTINES ARE USED :
C        1. ADVEC1 : GET FILE NAMES, IMAGE SECTION TO PROCESS, WINDOW SIZES
C           AND SHIFT FACTOR. IT ALSO CREATES THE STORAGE FILE XXXVEC.DAT .
C
C        2. ADVEC2 : USES THE INFORMATION PROVIDED BY ADVEC1 TO CREATE A
C           16-BIT , NONDECIMATED IMAGE FOR THE SPECIFIED IMAGE AREA -
C           IF NOT ENOUGH VMA AVAILABLE TO LOAD THE ENTIRE AREA, THEN AS
C           LARGE A NUMBER OF LINES OF SPECIFIED NO. OF PIXELS AS POSSIBLE
C           WILL BE LOADED AND FURTHER VISITS TO ADVEC2 WILL BE NECESSARY TO
C           PROCESS THE ENTIRE AREA. IT LOADS THE SPECIFIED TWO FEATURES IN
C           TO THE CREATED IMAGE (IMAGE 4). THE SUBROUTINE ALSO CREATES THREE
C           8-BIT IMAGES (IMAGES 1,2 AND 3) AS DATA BUFFERS FOR THE TEMPLATE
C           AND SEARCH ARRAYS DURING THE COMPUTATION OF THE MCC, AND TO HOLD
C           AN ARRAY OF VARIANCES USED IN THE MCC COMPUTATION IN ADVEC4.
C
C        3. ADVEC3 : CALCULATES SUCCESSIVE POSITIONS OF THE WINDOWS AND READ
C           THE WINDOW DATA FROM IMAGE 4, COMPUTE AVERAGES AND VARIANCES,
C           CHECK FOR THE PRESENCE OF LAND AND CLOUD WITHIN THE WINDOWS AND
C           STORE THE WINDOW DATA AS COMPLEX NUMBERS IN IMAGES 1 AND 2. IT
C           ALSO STORES THE VARIANCES (DENOMINATOR IN THE MCC EQUATION) IN
C           IMAGE 3.
C
C       4. ADVEC4 : PERFORM TWO DIMENSIONAL FAST FOURIER TRANSFORMS ON THE
C          ARRAYS CREATED BY ADVEC3 AND CARRY OUT THE OTHER MANIPULATIONS
C          TO DERIVE THE VECTORS.
C
C       5. FFT1   : THE SUBROUTINE USED BY ADVEC4 TO PERFORM ONE DIMENSIONAL
C          FAST FOURIER TRANSFORMS.
C
C
C       WRITTEN BY J.J.AGENBAG  S.F.R.I.  APRIL 1990. IN FORTRAN 77.
```

```
C       FOR A DEC LSI 11/23 BASED IMAGE PROCESSING SYSTEM USING DIPIX
C       ARIES II SOFTWARE MODULES FOR CERTAIN OPERATIONS.
C
        PROGRAM ADVECT
C
        IMPLICIT COMPLEX(A),BYTE(B),CHARACTER*1(C),INTEGER*4(E)
        DIMENSION BAR(3),BFINAM(9,2)
        COMMON IFMASK(3),IDECIM(3),IFMAPS(1536),NUMWPL(4),ESTADR(4)
        CHARACTER*7 NAMMOD, NAMSUB , VECTFN*10
C
        ICYC=1              ! FIRST VISIT TO ADVEC2
        NVEC=0              ! COUNTER FOR NUMBER OF DERIVED VECTORS.
C
C       1.0 CALL ADVEC1 TO INPUT DATA .
C
        CALL ADVEC1(QQQ,BAR,BFINAM,IFLPR,ILLPR,IFPPR,ILPPR,
     +  VECTFN,ITWS,ISWS,ISHIFT,IERR,NAMMOD,NAMSUB)
        WRITE(6,15)BAR,BFINAM,IFLPR,ILLPR,IFPPR,ILPPR,VECTFN,ITWS,ISWS,
     +  ISHIFT
15      FORMAT(1X/' ADVECT 1 : BAR,BFINAM= ',3A1,3X,9A1,2X,9A1,/,
     +  ' IFLPR,ILLPR,IFPPR,ILPPR=',2I6,3X,2I6,/,
     +  ' VECTFN=',A10,'  ITWS,ISWS,ISHIFT=',3I5)
        IF(IERR.NE.1)GOTO 80     ! EXIT WITH ERROR.
C
C       2.0 CALL ADVEC2 TO CREATE THE IMAGE OR LOAD ANOTHER IMAGE SECTION.
C
20      CALL ADVEC2(ISWS,ICYC,BAR,BFINAM,IFLPR,ILLPR,IFPPR,ILPPR,
     +  QQQ,LIN1,LIN2,IERR,NAMMOD,NAMSUB,RRR,IWINDL)
        WRITE(6,30)ISWS,ICYC,BAR,BFINAM,IFLPR,ILLPR,IFPPR,ILPPR,
     +  LIN1,LIN2,IWINDL
30      FORMAT(1X,/,'ADVECT 2 : ISWS,ICYC=',2I4,/,
     +  ' BAR,BFINAM=',3A1,3X,9A1,3X,9A1,/,
     +  ' IFLPR,ILLPR,IFPPR,ILPPR=',2I6,2X,2I6,/,
     +  ' LIN1,LIN2,IWINDL=',2I6,3X,I6)
        IF(IERR.NE.1)GOTO 80
C
C       3.0 CALL ADVEC3 TO TRANSFER WINDOW DATA TO BUFFER IMAGES.
C
40      CALL ADVEC3(ISWS,ITWS,ISHIFT,IFPPR,ILLPR,ILPPR,LIN1,LIN2,
     +  QQQ,IFLAG,ISWSL,ISWSP,IERR,NAMMOD,NAMSUB,RRR,IWINDL,IWINDP)
        WRITE(6,50)ISWS,ITWS,ISHIFT,IFPPR,ILLPR,ILPPR,LIN1,LIN2,
     +  IFLAG,ISWSL,ISWSP,IWINDL,IWINDP
50      FORMAT(1X/' ADVECT 3 : ISWS,ITWS,ISHIFT =',3I6,/,
     +  ' IFPPR,ILLPR,ILPPR =',3I6,/,
     +  ' LIN1,LIN2,IFLAG =',3I6,/,
     +  ' ISWSL,ISWSP,IWINDL,IWINDP =',4I6)
        IF(IERR.NE.1)GOTO 80     ! EXIT WITH ERROR.
        IF(IFLAG.NE.0)GOTO 60    ! GO EXTRACT VECTOR.
        IF(LIN2.LT.ILLPR)THEN
           ICYC=ICYC+1
           GOTO 20               ! GO LOAD NEXT IMAGE SECTION.
         ELSE
           GOTO 100              ! END
        END IF
C
C       4.0 CALL ADVEC4 TO EXTRACT THE VECTOR.
C
60      CALL ADVEC4(ITWS,ISWS,ISWSL,ISWSP,QQQ,NVEC,IERR,NAMSUB,NAMMOD)
        WRITE(6,65)ITWS,ISWS,ISWSL,ISWSP,NVEC
65      FORMAT(1X/' ADVECT 4 : ITWS,ISWS,ISWSL,ISWSP,NVEC=',4I5,I3)
```

```
      IF(IERR.NE.1)GOTO 80
      IF((IWINDL+ISWS-1).LE.LIN2)GOTO 40       ! GO READ NEXT WINDOW.
      IF((IWINDL+ISWS-1).LE.ILLPR)THEN         ! GO LOAD NEXT IMAGE SECTION.
         ICYC=ICYC+1
         GOTO 20
        ELSE
         GOTO 100
      END IF
C
C     5.0 END
C
80    WRITE(5,90)IERR,NAMSUB,NAMMOD
90    FORMAT(1X//' EXITTING WITH ERROR',I4,'  IN SUBROUTINE ',A7,/,
     +' MODULE ',A7)
C
100   WRITE(5,120)BFINAM,VECTFN,NVEC
120   FORMAT(1X//' IMAGE FEATURES ',9A1,' AND ',9A1,/,
     +' NUMBER OF VECTORS SAVED IN FILE ',A10,' =',I6)
      CLOSE(4)
      CALL EXIT
      END
```

```
C
C       SUBROUTINE ADVEC1.FTN
C       A SUBROUTINE CALLED BY PROGRAM ADVECT WHICH PERFORMS AUTOMATIC
C       FEATURE TRACKING TO DERIVE SURFACE ADVECTION VECTORS.
C       ADVEC1 OBTAINS FILE NAMES, AREA TO PROCESS, WINDOW SIZES AND SHIFT
C       A FILE , XXXVEC.DAT, IS CREATED TO STORE THE VECTOR START AND END
C       LINE/PIXEL COORDINATES ( XXX= AREA CODE FOR IMAGE).
C
C       WRITTEN BY J.J.AGENBAG  S.F.R.I.  APRIL 1989. IN FORTRAN 77.
C       FOR A DEC LSI 11/23 BASED IMAGE PROCESSING SYSTEM USING DIPIX
C       ARIES II SOFTWARE MODULES FOR CERTAIN OPERATIONS.
C
        SUBROUTINE ADVEC1(QQQ,BAR,BFINAM,IFLPR,ILLPR,IFPPR,ILPPR,
     +  VECTFN,ITWS,ISWS,ISHIFT,IERR,NAMMOD,NAMSUB)
C
        IMPLICIT COMPLEX(A),BYTE(B),CHARACTER*1(C),INTEGER*4(E)
        DIMENSION BAR(3),BFINAM(9,2),BFOUND(9),IBH(2),IBL(2)
        CHARACTER NAMMOD*7,NAMSUB*7,VECTFN*10
C
        PARAMETER (LUNIMG=1)
C
C
        NAMSUB='ADVEC1 '
        IERR=1
        VECTFN='***VEC.DAT'
C
C       1.0 GET IMAGE FILE NAMES. CHECK WHETHER THEY EXIST. GET DIMENSIONS.
C
10      WRITE(5,20)
20      FORMAT(1X/' ENTER 3-CHAR. AREA CODE FOR THE IMAGE TO BE ',
     +  'PROCESSED :',$)
        READ(5,40)VECTFN(1:3)
40      FORMAT(A3)
        DO 60 I=1,3                ! CONSTRUCT FILE NAME.
          WRITE(CHAR,50)VECTFN(I:I)
50        FORMAT(A1)
          READ(CHAR,50)BAR(I)
          BFINAM(I,1)=BAR(I)
          BFINAM(I,2)=BAR(I)
60      CONTINUE
        DO 80 I=4,5
          BFINAM(I,1)='F'
          BFINAM(I,2)='F'
80      CONTINUE
        DO 100 I=6,9
          BFINAM(I,1)='*'
          BFINAM(I,2)='*'
100     CONTINUE
        CALL ZFSEEK(BFINAM(1,1),1,QQQ,IERR,ISIZE,IBH(1),IBL(1),BFOUND)
        IF(IERR.EQ.1)GOTO 140    ! FILES WITH THIS AREA CODE EXIST.
        IF(IERR.EQ.-261)THEN     ! NO FILES WITH THIS CODE.
           WRITE(5,120)BAR
120        FORMAT(1X//' NO FILES WITH AREA CODE ',3A1,' FOUND.',
     +     ' RESPECIFY PLEASE.')
           GOTO 10
          ELSE
           NAMMOD='ZFSEEK1'      ! SOME OTHER ERROR.
           GOTO 1000
        END IF
```

```
C
C       AREA CODE EXIST. GET DIMENSIONS.
C
140     CALL ZHRDIM(BFINAM(1,1),LUNIMG,QQQ,IERR,NLIN,NPIX,IFP,IFL,ILP,
     +  ILL)
        IF(IERR.EQ.1)GOTO 160
        NAMMOD='ZHRDIM '
        GOTO 1000
C
C       INPUT FILE ID'S. CHECK.
C
160     WRITE(5,170)
170     FORMAT(1X/' ENTER THE FILE ID S FOR THE TWO FEATURE FILES',/,
     +  ' ON WHICH FEATURE TRACKING IS TO BE PERFORMED. THE FIRST ',/,
     +  ' WILL BE THE TEMPLATE IE. THE FIRST IMAGE IN TIME.')
        DO 300 I=1,2
180       WRITE(5,190)I
190       FORMAT(1X/' FOR FEATURE NO.',I2,' ENTER 4-CHAR FILE ID :',$)
          READ(5,200)(BFINAM(J,I),J=6,9)
200       FORMAT(4A1)
          CALL ZFSEEK(BFINAM(1,I),1,QQQ,IERR,ISIZE,IBH(I),IBL(I),BFOUND)
          IF(IERR.EQ.1)GOTO 300     ! FILE EXIST.
          IF(IERR.EQ.-261)THEN      ! FILE DO NOT EXIST.
             WRITE(5,240)(BFINAM(J,I),J=1,9)
240          FORMAT(1X/' FILE ',9A1,' DO NOT EXIST. RESPECIFY PLSE.')
             GOTO 180
            ELSE
             NAMMOD='ZFSEEK2'       ! SOME OTHER ERROR.
             GOTO 1000
          END IF
300     CONTINUE
C
C       2.0 GET AREA TO BE PROCESSED.
C
        WRITE(5,320)IFL,ILL,IFP,ILP
320     FORMAT(1X//' FILE DIMENSIONS ARE :',/,
     +  ' FIRST LINE =',I5,'  LAST LINE =',I5,/,
     +  ' FIRST PIXEL=',I5,'  LAST PIXEL=',I5)
330     WRITE(5,340)
340     FORMAT(1X/' ENTER FIRST AND LAST LINE TO PROCESS:',$)
        READ(5,*)IFLPR,ILLPR
        IF(IFLPR.LT.IFL.OR.ILLPR.GT.ILL.OR.ILLPR.LT.IFLPR) GOTO 330
350     WRITE(5,360)
360     FORMAT(1X/' ENTER FIRST AND LAST PIXEL TO PROCESS :',$)
        READ(5,*)IFPPR,ILPPR
        IF(IFPPR.LT.IFP.OR.ILPPR.GT.ILP.OR.ILPPR.LT.IFPPR)GOTO 350
        NUMPIX=ILPPR-IFPPR+1
        NPIX=2*INT(FLOAT(NUMPIX)/2.0)
        IF(NPIX.NE.NUMPIX)ILPPR=ILPPR+1
        IF(ILPPR.GT.ILP)ILPPR=ILPPR-2           ! PIXELS MUST BE EVEN.
        NUMPIX=ILPPR-IFPPR+1
C
C       3.0 ENTER THE WINDOW SIZES AND WINDOW SHIFT DISTANCE.
C
        WRITE(5,380)
380     FORMAT(1X//' TEMPLATE AND SEARCH WINDOWS WILL BE N1XN1 AND',/,
     +  ' N2XN2 PIXELS RESPECTIVELY. N2 MUST BE 4,8,16,32 OR 64',/,
     +  ' N1 MUST BE EVEN AND LESS THAN N2.')
390     WRITE(5,400)
400     FORMAT(1X/' ENTER N1 AND N2 :',$)
```

```
      READ(5,*)ITWS,ISWS
      ITW=2*INT(FLOAT(ITWS)/2.0)
      ISW=2*INT(FLOAT(ISWS)/2.0)
      IF(ITWS.GE.ISWS.OR.ITWS.NE.ITW.OR.ISWS.NE.ISW)GOTO 390
      IF(ITWS.GT.64.OR.ISWS.GT.64)GOTO 390
      IWINDL=0
410   WRITE(5,420)
420   FORMAT(1X//' ENTER THE WINDOW SHIFT DISTANCE FOR SUCCESSIVE',/,
     +' VECTOR CALCULATIONS ( SHOULD BE >=1) :',$)
      READ(5,*)ISHIFT
      IF(ISHIFT.LT.1)GOTO 410
C
C     4.0 CREATE THE STORAGE FILE FOR VECTOR COORDINATES.
C
      OPEN(4,FILE=VECTFN,STATUS='NEW')
C
C     6.0 RETURN
C
1000  RETURN
      END
```

```
C
C       SUBROUTINE ADVEC2.FTN
C       A SUBROUTINE CALLED BY PROGRAM ADVECT WHICH PERFORMS AUTOMATIC
C       FEATURE TRACKING TO DERIVE SURFACE ADVECTION VECTORS.
C       ADVEC2 USES THE INFORMATION SUPPLIED BY ADVEC1 TO CREATE .
C        A. TWO BLANK 8-BIT IMAGES WITH DIMENSIONS N2 LINES BY N2*8 PIXELS
C           TO HOLD THE TEMPLATE AND SEARCH WINDOW DATA AS 8-BYTE COMPLEX
C           NUMBERS ( SEARCH WINDOW = N2 X N2 )
C        B. A BLANK 8-BIT IMAGE WITH DIMENSIONS N2 LINES BY N2*4 PIXELS TO
C           HOLD A REAL N2XN2 ARRAY CONTAINING SQRT(TEMPLATE VARIANCE * SEARCH
C           WINDOW VARIANCE). THIS ARRAY IS FILLED BY ADVEC3 AND USED BY
C           ADVEC4 IN THE COMPUTATION OF THE MAX. CROSS CORRELATION.
C        C. A 16-BIT NON-DECIMATED IMAGE CONTAINING THE TWO SPECIFIED FEATURES
C
C       IF NOT ENOUGH VMA AVAILABLE , ADVEC2 WILL LOAD AS LARGE A  NUMBER
C       OF LINES ( OF SPECIFIED NUMBER OF PIXELS) AS WILL FIT INTO AVAILABLE
C       VMA. THE ROUTINE WILL THEN BE VISITED AGAIN TO PROCESS THE REMAINING
C       PART(S) OF THE IMAGE.
C
C       WRITTEN BY J.J.AGENBAG  S.F.R.I.  APRIL 1989. IN FORTRAN 77.
C       FOR A DEC LSI 11/23 BASED IMAGE PROCESSING SYSTEM USING DIPIX
C       ARIES II SOFTWARE MODULES FOR CERTAIN OPERATIONS.
C
        SUBROUTINE ADVEC2(ISWS,ICYC,BAR,BFINAM,IFLPR,ILLPR,IFFPR,ILPPR,
     +  QQQ,LIN1,LIN2,IERR,NAMMOD,NAMSUB,RRR,IWINDL)
C
        IMPLICIT COMPLEX(A),BYTE(B),CHARACTER*1(C),INTEGER*4(E)
        DIMENSION BAR(3),BFINAM(9,2),BTNAME(13),BIXD(2),BNFEAT(2),
     +  BFDPTH(2),BDUM(2),BNTFIL(2),BNTHMS(2),BBB(3)
        COMMON IFMASK(3),IDECIM(3),IFMAPS(1536),NUMWPL(4),ESTADR(4)
        CHARACTER NAMMOD*7,NAMSUB*7
C
        DATA BTNAME/'I','M','I','S','F','E','V','M','A','.','S','F','F'/
        PARAMETER (LUNDD=2)
C
        EQUIVALENCE (BIXD(1),IDEPTH),(BNFEAT(1),IFEAT),(BDUM(1),IFD),
     +  (BNTFIL(1),NTFIL),(BNTHMS(1),NTHMS)
C
        NAMSUB='ADVEC2 '
        IERR=1
        IF(ICYC.NE.1)GOTO 180
C
C       1.0 ATTACH VIDEO SUBSYSTEM AND CREATE THE TWO DATA BUFFERS (IMAGE 1
C           AND IMAGE 2) FOR THE TEMPLATE AND SEARCH WINDOWS RESPECTIVELY.
C
        CALL ZIGVID(LUNDD,QQQ,IERR)
C       WRITE(5,50)LUNDD,IERR
C50     FORMAT(1X/' ADVEC2 1: ZIGVID : LUNDD=',I3,'  IERR=',I5)
        IF(IERR.EQ.1)GOTO 100
        NAMMOD='ZIGVID '
        GOTO 1000                   ! EXIT WITH ERROR.
C
100     CALL ZITOPL(BTNAME,QQQ,IERR)
C       WRITE(5,105)BTNAME,IERR
C105    FORMAT(1X/' ADVEC2 2 : ZITOPL : BTNAME=',13A1,' IERR=',I5,/)
        IF(IERR.EQ.1)GOTO 120
        NAMMOD='ZITOPL '
        GOTO 1000                   ! EXIT WITH ERROR
C
```

```
120     IDEPTH =7          ! IXEL DEPTH. BIXD(1) FROM EQUIVALENCE.
        IFEAT=1            ! NUMBER OF FEATURES. BNFEAT FROM EQUIVALENCE.
        IFD=8              ! BITS PER FEATURE.   BFDPTH  ,,      ,,
        BFDPTH(1)=BDUM(1)
        NTFIL=0            ! NUMBER OF THEME FILES. BNTFIL ,,    ,,
        NTHMS=0            ! NUMBER OF THEMES.   BNTHMS(1) ,,    ,,
        NUMLIN=ISWS        ! NUMBER OF DISPLAY LINES ( = SEARCH WINDOW DIMENSION)
        NUMPIX=ISWS*8      ! NUMBER OF DISPLAY PIXELS. THE IMAGE WILL STORE
                           ! COMPLEX NUMBERS IE. 8 BYTES PER PIXEL.
C
130     DO 160 I=1,3       ! CREATE THE 3 DATA BUFFERS (IMAGES 1,2 AND 3)
          IMGNUM=I
          J=(I-1)*512+1 ! MAPPING TABLE IMAGE 1= 1 TO 512, IMAGE 2= 513-1024
          IF(I.EQ.3)NUMPIX=ISWS*4
          CALL ZIALOC(BIXD(1),NUMLIN,NUMPIX,BNFEAT(1),BFDPTH(1),
     +    BNTFIL(1),BNTHMS(1),'P','N','C',QQQ,IERR,IMGNUM,ESTADR(I),
     +    RRR,BBB,II,JJ)
C         WRITE(6,135)I,BIXD(1),NUMLIN,NUMPIX,BNFEAT(1),BFDPTH(1),
C    +    BNTFIL(1),BNTHMS(1),IMGNUM,ESTADR(I),RRR,BBB,II,JJ
C135      FORMAT(1X/' ADVEC2 3: ZIALOC : I=',I3,/,
C    +    ' BIXD(1),NUMLIN,NUMPIX,BNFEAT(1)=',O3,2I7,2X,O3,/,
C    +    ' BFDPTH(1),BNTFIL(1),BNTHMS(1),IMGNUM=',3(O3,1X),I6,/,
C    +    ' ESTADR,RRR,BBB,II,JJ=',I8,F8.4,2X,3A1,2I6)
          IF(IERR.EQ.1)GOTO 140
           NAMMOD='ZIALOC '
           GOTO 1000                    ! EXIT WITH ERROR.
C
140       CALL ZMFMAP(IMGNUM,1,255,0,'E',QQQ,IERR,IFMASK(I),IFMAPS(J),
     +    IXD,IDECIM(I),NUMWPL(I),ESTADR(I))    ! GET MAPPING DATA.
C         WRITE(6,150)I,J,IFMASK(I),IXD,IDECIM(I),NUMWPL(I),ESTADR(I)
C150      FORMAT(1X/' ADVEC2 4 : ZMFMAP : I,J,IFMASK(I)=',3I6,/,
C    +    ' IXD,IDECIM(I),NUMWPL(I),ESTADR(I)=',3I5,I8)
          IF(IERR.EQ.1)GOTO 160
           NAMMOD='ZMFMAP '
           GOTO 1000                    ! EXIT WITH ERROR.
160     CONTINUE
        GOTO 190           ! SKIP IMAGE DELETE REQUIRED FOR MULTIPLE LOAD OF
                           ! LARGE IMAGES.
C
C       2.0 DELETE IMAGE NUMBER FOUR.
C
180     CALL ZIDELE(4,'C',QQQ,IERR)
C       WRITE(6,185)IERR
C185    FORMAT(1X/' ADVEC2 5 : ZIDELE : IERR=',I5)
        IF(IERR.EQ.1)GOTO 190
         NAMMOD='ZIDELE2'
         GOTO 1000                      ! EXIT WITH ERROR.
C
C       3.0 DETERMINE HOW MUCH VIDEO MEMORY IS FREE AND CREATE THE IMAGE
C           TO BE PROCESSED - IE. IMAGE NO.3
C
C
190     CALL ZIFREE(QQQ,IERR,EVMTOT,EVMCON)      ! HOW MUCH FREE VMA ?
C       WRITE(6,195)IERR,EVMTOT,EVMCON
C195    FORMAT(1X/' ADVEC2 6 : ZIFREE : IERR=',I6,/,
C    +  ' EVMTOT,EVMCON=',2I8)
        IF(IERR.EQ.1)GOTO 200
        NAMMOD='ZIFREE '
        GOTO 1000                  ! EXIT WITH ERROR.
C
```

```
C       3.1 CALCULATE HOW MANY LINES MAY BE LOADED AS 16-BIT NONDECIMATED.
C
200     RNL=FLOAT(EVMCON-152)/FLOAT(ILPPR-IFPPR+1)
        IF(RNL.GT.10000)RNL=10000.0
        NUMBLN=INT(RNL)
        IF(ICYC.EQ.1)THEN                    ! FIRST VISIT TO ADVEC1.
           LIN1=IFLPR              ! FIRST LINE TO LOAD.
           LIN2=ILLPR              ! LAST   ,,  ,,  ,,
         ELSE
           LIN1=IWINDL           ! NEXT LINE TO PROCESS. FROM ADVEC2
           LIN2=ILLPR
        END IF
        IF(NUMBLN.LT.(LIN2-LIN1+1))LIN2=LIN1+NUMBLN-1
        NUMLIN=LIN2-LIN1+1
C
C       3.2 CREATE THE IMAGE AND LOAD THE FEATURES.
C
        RDEC=1.0
        IDEPTH=15          ! IXEL DEPTH. BIXD(1) FROM EQUIVALENCE
        IFEAT=2            ! NUMBER OF FEATURES. BNFEAT FROM EQUIVALENCE.
        BFDPTH(2)=BFDPTH(1) ! BITS TO ALLOCATE TO FEATURE 2.
        NUMPIX=ILPPR-IFPPR+1
        NUMWPL(4)=NUMPIX
        IMGNUM=4
        CALL ZICREA(BAR,BIXD(1),LIN1,IFPPR,NUMLIN,NUMPIX,BNFEAT(1),
     +  BFDPTH,BNTFIL(1),BNTHMS(1),'P','N','N',1.0,QQQ,IERR,
     +  IMGNUM,ESTADR(4),RRR,RDEC)
C       WRITE(6,210)BAR,BIXD(1),LIN1,IFPPR,NUMLIN,NUMPIX,BNFEAT(1),
C    +  BFDPTH,BNTFIL(1),BNTHMS(1),IMGNUM,ESTADR(4),RDEC
C210    FORMAT(1X/' ADVEC2 7 : ZICREA : BAR,BIXD(1)= ',3A1,2X,O3,/,
C    +  ' LIN1,IFPPR,NUMLIN,NUMPIX=',4I6,/,
C    +  ' BNFEAT(1),BFDPTH,BNTFIL(1),BNTHMS(1)=',5O5,/,
C    +  ' IMGNUM,ESTADR(4),RDEC =',I3,I8,F8.4)
        IF(IERR.EQ.1)GOTO 220
        NAMMOD='ZICREA '
        GOTO 1000                  ! EXIT WITH ERROR.
C
220     ISATUR=255
        ICUTOF=0
        IWINDL=0           ! SET TO ZERO AS FLAG FOR ADVEC3
        DO 300 I=1,2
          CALL ZIFEAT(IMGNUM,I,'O','F',QQQ,IERR,RRR,ISATUR,
     +    ICUTOF,BFINAM(1,I))
C         WRITE(6,230)IMGNUM,I,IERR,ISATUR,ICUTOF,(BFINAM(J,I),J=1,9)
C230      FORMAT(1X/' ADVEC2 8: ZIFEAT : IMGNUM,I,IERR=',3I6,/,
C    +    ' ISATUR,ICUTOF,BFINAM(J,I)=',2I5,3X,9A1)
          IF(IERR.EQ.1)GOTO 300
          NAMMOD='ZIFEAT '
          GOTO 1000                ! EXIT WITH ERROR.
300     CONTINUE
C
C       4.0 RETURN
C
1000    RETURN
        END
```

```
C
C       SUBROUTINE ADVEC3.FTN
C       A SUBROUTINE CALLED BY PROGRAM ADVECT WHICH PERFORMS AUTOMATIC
C       FEATURE TRACKING TO DERIVE SURFACE ADVECTION VECTORS.
C       ADVEC3 CALCULATES THE LOCATIONS OF SUCCESSIVE TEMPLATE AND SEARCH
C       WINDOWS IN VMA AND READ THE DATA, CONVERT INTO COMPLEX NUMBERS.
C       THE TEMPLATE DATA ARE STORED IN IMAGE 1 AND THE SEARCH WINDOW DATA
C       IN IMAGE 2. THE ROUTINE ALSO COMPUTES WINDOW VARIANCES AND STORE
C       THE SQRT(TEMPLATE VARIANCE X SEARCH WINDOW VARIANCE) IN IMAGE 3.
C
C       CLOUD AND LAND MUST BE SET TO 255 IN THE FEATURES. ADVEC3 REJECTS
C       ANY WINDOW POSITION WHERE MORE THAN 1% OF PIXELS IN EITHER
C       THE TEMPLATE OR SEARCH IMAGE CONTAIN LAND OR CLOUD.
C       THE MAXIMUM WINDOW SIZE HANDLED BY THIS ROUTINE IS 64 X 64.
C
C       WRITTEN BY J.J.AGENBAG  S.F.R.I.  APRIL 1990. IN FORTRAN 77.
C       FOR A DEC LSI 11/23 BASED IMAGE PROCESSING SYSTEM USING DIPIX
C       ARIES II SOFTWARE MODULES FOR CERTAIN OPERATIONS.
C
C
      SUBROUTINE ADVEC3(ISWS,ITWS,ISHIFT,IFPPR,ILLPR,ILPPR,LIN1,LIN2,
     +  QQQ,IFLAG,ISWSL,ISWSP,IERR,NAMMOD,NAMSUB,RRR,IWINDL,IWINDP)
C
      IMPLICIT COMPLEX(A),BYTE(B),CHARACTER*1(C),INTEGER*4(E)
      DIMENSION ARR1(64),IARR(256),RARR(64),RARR1(2,64),
     +  IARR1(64,64),BCON(2),BARR(2,64)
      COMMON IFMASK(3),IDECIM(3),IFMAPS(1536),NUMWPL(4),ESTADR(4)
      CHARACTER*7 NAMMOD,NAMSUB
      EQUIVALENCE (ARR1(1),IARR(1),RARR(1),BARR(1,1)),
     +  (BCON(1),ICON)
C
C     WRITE(6,20)ISWS,ITWS,ISHIFT,IFPPR,ILLPR,ILPPR,LIN1,LIN2,
C    +  IFLAG,ISWSL,ISWSP,IWINDL,IWINDP,IFMASK,IDECIM,NUMWPL,ESTADR
C20   FORMAT(1X//' ADVEC3 1: ISWS,ITWS,ISHIFT =',3I6,/,
C    +  ' IFPPR,ILLPR,ILPPR,LIN1,LIN2=',5I6,/,
C    +  ' IFLAG,ISWSL,ISWSP,IWINDL,IWINDP=',5I6,/,
C    +  ' IFMASK,IDECIM =',3I4,3X,3I5,/,
C    +  ' NUMWPL,ESTADR=',4I5,4I8)
      IERR=1
      ICON=0
      NAMSUB='ADVEC3 '
C
C       1.0 SET WINDOWS.
C
      IF(IWINDL.EQ.0)THEN                ! A NEW IMAGE SECTION.
        IWINDL=LIN1                 ! SET FIRST SEARCH WINDOW TO TOP LEFT CORNER
        IWINDP=IFPPR                ! OF LOADED IMAGE SECTION.
      END IF
C
50    ISWSL=IWINDL                  ! SEARCH WINDOW START LINE.
      ISWSP=IWINDP                  !  ,,   ,,     ,,     PIXEL
      ISWLL=IWINDL+ISWS-1           ! LAST LINE IN SEARCH WINDOW
      ISWLP=IWINDP+ISWS-1           !  ,,  PIXEL ,,  ,,     ,,
      ITWSL=IWINDL+(ISWS-ITWS)/2    ! START LINE IN TEMPLATE WINDOW.
      ITWLL=ITWSL+ITWS-1            ! LAST  ,,   ,,   ,,      ,,
      ITWSP=IWINDP+(ISWS-ITWS)/2    ! START PIXEL IN TEMPLATE WINDOW
      ITWLP=ITWSP+ITWS-1            !  LAST   ,,  ,,    ,,      ,,
      ITWLIM=INT(FLOAT(ITWS*ITWS)*.01+0.5)     ! LAND/CLOUD LIMIT TEMPLATE
      ISWLIM=INT(FLOAT(ISWS*ISWS)*.01+0.5)  ! SEARCH WINDOW LIMIT LAND/CLOUD
```

```
C
C       WRITE(6,55)ISWSL,ISWSP,ISWLL,ISWLP,ITWSL,ITWLL,ITWSP,ITWLP,
C     +  ITWLIM,ISWLIM
C55     FORMAT(1X/' ADVEC3 2 : ISWSL,ISWSP,ISWLL,ISWLP=',4I6,/,
C     +  ' ITWSL,ITWLL,ITWSP,ITWLP= ',4I6,/,
C     +  ' ITWLIM,ISWLIM=',2I6)
C
C       2.0 READ WINDOW DATA FROM IMAGE 4 AND STORE AS INTEGERS IN IARR1()
C           FIRST DO TEMPLATE WINDOW THEN SEARCH WINDOW DATA. IN THE PROCESS
C           COMPUTE THE MEAN PIXEL COUNT FOR EACH WINDOW AND CHECK FOR THE
C           PRESENCE OF CLOUD AND/OR LAND OR OTHER NON-DATA AREAS MASKED BY
C           VALUES OF 255. IF SUCH PIXELS CONSTITUTE MORE THAN 1% OF THE
C           WINDOW THEN ABORT THE WINDOW. PIXEL COUNTS ARE CONVERTED AND
C           STORED AS COMPLEX NUMBERS IN IMAGES 1 AND 2.
C
        E2WPL=NUMWPL(4)*2          ! CONSTANT USED IN READING IMAGE DATA.
        NP=ITWS                ! SET UP VARIABLES FOR PROCESSING THE TEMPLATE
        IP=ITWSP
        IB=2
        L1=ITWSL
        L2=ITWLL
        ILIM=ITWLIM
        DO 300 IM=1,2                                  ! 1=TEMPLATE , 2= SEARCH WINDO
          IFM=1+(IM-1)*512                             ! CONSTANTS USED IN READING
          EPDIF=(IP-IFPPR)*2                           ! FROM AND WRITING TO
          EVA1=ESTADR(4)-LIN1*E2WPL+EPDIF              ! IMAGE MEMORY
          NUMLC=0                              ! COUNTER FOR NON-DATA PIXELS
          SUM1=0.0                             ! SUM FOR WINDOW AVERAGE
C
C         2.1 READ AND EXTRACT PIXELS AS INTEGERS. STORE IN IARR1().
C
C         WRITE(6,95)IM,E2WPL,NP,IP,L1,L2,ILIM,IFM,EPDIF,EVA1
C95       FORMAT(1X/' ADVEC3 3 : IM,E2WPL,NP,IP=',I2,I8,2I6,/,
C     +     ' L1,L2,ILIM,IFM=',4I6,/,
C     +     ' EPDIF,EVA1 =',2I8)
          K=0
          DO 140 L=L1,L2                       ! READ LINE BY LINE
            K=K+1                              ! INDEX TO IARR1()
            EVADDR=EVA1+L*E2WPL
            CALL ZMRVMA(EVADDR,NP,QQQ,IERR,IARR)
            IF(IERR.EQ.1)GOTO 100
             NAMMOD='ZMRVMA '
             GOTO 1000                         ! EXIT WITH ERROR
C
100         DO 120 I=1,NP                      ! EXTRACT PIXEL COUNTS FROM LINE L
              BCON(1)=BARR(IB,I)               ! CONVERT TO INTEGER VIA EQUIVALENCE
              IF(ICON.EQ.255)THEN              ! CHECK FOR LAND/CLOUD
                NUMLC=NUMLC+1                  ! INCREMENT LAND/CLOUD COUNT
                IF(NUMLC.GT.ILIM)GOTO 800   ! TOO MUCH. ABORT.
                ICON=0
              END IF
              IARR1(I,K)=ICON                  ! STORE AS INTEGER.
              SUM1=SUM1+FLOAT(ICON)            ! SUM FOR CALCULATION OF AVERAGE.
120           CONTINUE
C             WRITE(6,130)L,K,(IARR1(I,K),I=1,NP)
C130          FORMAT(1X,2I6,16I4,3(/,1X,12X,16I4))
140       CONTINUE
C
C         2.2 TRANSFER THE DATA FROM IARR1() TO IMAGE 1 OR 2 AS COMPLEX
C             NUMBERS. IN THE PROCESS MUST : (1) SUBTRACT THE AVERAGE
```

```
C             (2) CONVERT THE PIXEL COUNT TO DEG.C/NM
C             (3) COMPUTE THE WINDOW VARIANCE ( FOR THE TEMPLATE ONLY ).
C
          RMEAN=SUM1/FLOAT(NP*NP-NUMLC)          ! THE WINDOW AVERAGE.
          RFACT=1.0                              ! CONVERSION FACTOR
          SUM1=0.0
          IF(IM.EQ.1)THEN                  ! FILL TRANSFER BUFFER WITH
            A0=CMPLX(0,0)                  ! COMPLEX ZERO. NO NEED TO DO
            DO 160 I=1,ISWS                ! THIS FOR THE SEARCH ARRAY.
              ARR1(I)=A0
160         CONTINUE
          END IF
          L=L1-ISWSL                       ! ROW NUMBER IN STORAGE IMAGE 1 OR 2
C         WRITE(6,165)IM,SUM1,NUMLC,RMEAN,L
C165      FORMAT(1X/' ADVEC3 5: IM,SUM1,NUMLC=',I3,F10.0,I5,/,
C    +    ' RMEAN,L=',F10.4,3X,I6,//' ADVEC3 6 :',/)
          DO 200 J=1,NP                    ! TRANSFER ROWS
            L=L+1
            K=IP-ISWSP                     ! PIXEL NO. IN STORAGE IMAGE 1 OR 2
            DO 180 I=1,NP                  ! ELEMENTS WITHIN THE ROW.
              K=K+1
              R1=(FLOAT(IARR1(I,J))-RMEAN)*RFACT
              ARR1(K)=CMPLX(R1,0.0)
              SUM1=SUM1+R1*R1              ! SUM FOR TEMPLATE VARIANCE
180         CONTINUE
            CALL ZMWLIN(IARR,L,IFMASK(IM),IFMAPS(IFM),7,IDECIM(IM),
     +      NUMWPL(IM),ESTADR(IM))         ! WRITE LINE L TO IMAGE 1 OR 2
C
C           WRITE(6,185)L,(ARR1(I),I=1,ISWS)
C185        FORMAT(1X,I4,5(F8.3,F5.2,2X),50(/,1X,4X,5(F8.3,F5.2,2X)))
C
200       CONTINUE
          IF(IM.EQ.1)VART=SQRT(SUM1/FLOAT(NP*NP))   ! SQRT OF TEMPLATE VARIANCE
C
          NP=ISWS                  ! SET UP
          IP=ISWSP                 ! VARIABLES
          IB=1
          L1=ISWSL                 ! FOR
          L2=ISWLL                 ! SEARCH
          ILIM=ISWLIM              ! WINDOW
300     CONTINUE
C       WRITE(6,305)VART
C305    FORMAT(1X//' ADVEC3 7: VART=',F9.4,/,
C    +  ' ADVEC3 8 : K AND RARR(I),I=1,ISWS :',/)
C
C       3.0 COMPUTE AN ARRAY OF VARIANCES FOR THE SEARCH ARRAY - ONE VALUE
C       FOR EACH POSITION OF THE TEMPLATE ARRAY WITHIN THE SEARCH ARRAY.
C       STORE IN IMAGE 3. THE VALUE IS WRITTEN IN THE BOTTOM RIGHT PIXEL OF
C       THE 4 PIXELS SURROUNDING THE TEMPLATE CENTRE ( THE TEMPLATE HAS EVEN
C       DIMENSIONS HENCE NO CENTRAL PIXEL EXIST.)
C       THE PROCEDURE CONSISTS OF 'POSITIONING' THE TEMPLATE IN THE TOP LEFT
C       CORNER OF THE SEARCH WINDOW. AT THIS POSITION  SUM(X) AND SUM(X**2)
C       ARE OBTAINED FOR ALL COLUMNS(OVER THE TEMPLATE). THESE ARE STORED IN
C       RARR1(1,I) AND RARR1(2,I). THE VARIANCE IS COMPUTED FROM THE COLUMN
C       SUMS. THE TEMPLATE IS THEN SHIFTED ONE COLUMN TO THE RIGHT. THE SUMS
C       IN RARR1() SHIFTED LEFT AND SUMS CALCULATED FOR THE NEW RIGHT HAND
C       COLUMN. A SIMILAR PROCEDURE IS USED TO SHIFT DOWN WHEN THE RIGHT-MOST
C       POSITION IS REACHED. THEN SHIFTING IS TOWARDS THE LEFT. ETC.
C
C
```

```
      RN=FLOAT(ITWS*ITWS)
C
C
C     3.1 IF THIS IS THE FIRST WINDOW BEING PROCESSED, THEN FILL OUT THE
C         STORAGE IMAGE (IMAGE 3) WITH VALUE 1.0
C
      DO 310 I=1,ISWS
        RARR(I)=1.0
310   CONTINUE
      IF(IWINDL.EQ.LIN1.AND.IWINDP.EQ.IFPPR)THEN      ! FILL IMAGE 3 WITH 1.(
        DO 320 I=1,ISWS
          CALL ZMWLIN(IARR,I,IFMASK(3),IFMAPS(1025),7,IDECIM(3),
     +      NUMWPL(3),ESTADR(3))                      ! WRITE TO IMAGE 3.
320     CONTINUE
      END IF
C
C     3.2 COMMENCE COMPUTATION OF VARIANCE. START IN THE TOP LEFT CORNER
C         AND SHIFT TO THE RIGHT; THEN ONE ROW DOWN AND TOWARDS THE LEFT, ET(
C
      I1=1
      I2=ISWS-ITWS+1
      I3=1
      DO 600 J=1,ISWS-ITWS+1   ! TOP LEFT ROW POSITION FOR THE TEMPLATE.
        K=J+ITWS/2             ! ROW POSITION FOR VARIANCE IN IMAGE 3
C
C       3.2.1 INITIAL TEMPLATE POSITION.
C
        IF(J.EQ.1)THEN
          L=1+ITWS/2           ! COLUMN POSITION FOR STORING VARIANCE
          DO 330 M=1,ITWS              ! ZERO THE COLUMN SUMS
            RARR1(1,M)=0.0             ! ACCUMULATOR FOR X
            RARR1(2,M)=0.0             !       ,,      ,, X**2
330       CONTINUE
          SUM1=0.0             ! USE AS OVERALL SUM(X)
          SUM2=0.0             ! OVERALL SUM(X**2)
C
          DO 360 M=1,ITWS              ! ALL THE COLUMNS.
            DO 340 N=1,ITWS            ! ELEMENTS WITHIN THE COLUMNS.
              R1=(FLOAT(IARR1(M,N))-RMEAN)*RFACT     ! CONVERT PIXEL COUNT
              RARR1(1,M)=RARR1(1,M)+R1               ! SUM(X)
              RARR1(2,M)=RARR1(2,M)+R1*R1            ! SUM(X**2)
340         CONTINUE
            SUM1=SUM1+RARR1(1,M)     ! OVER ALL SUM(X)
            SUM2=SUM2+RARR1(2,M)     ! OVER ALL SUM(X**2)
360       CONTINUE
          RARR(L)=SQRT(SUM2/RN-(SUM1/RN)**2.0)*VART  ! VARIANCE FACTOR
C         WRITE(6,365)J,L,RARR(L),SUM1,SUM2,((RARR1(J1,J2),
C    +      J2=1,ITWS),J1=1,2)
C365      FORMAT(1X/' J=',I3,' L=',I3,'  RARR(L),SUM1,SUM2=',3F9.4,/,
C    +      (1X,4X,8F9.4,/))
C
C           3.2.2 START A NEW ROW.
C           SUBTRACT PREVIOUS TOP ROW VALUES FROM THE COLUMN SUMMATIONS
C           AND ADD THOSE FROM THE NEW BOTTOM ROW.
C
          ELSE
           SUM1=0.0              ! OVERALL SUM(X)
           SUM2=0.0              ! OVERALL SUM(X**2)
           L=I1+ITWS/2           ! COLUMN POSITION FOR WRITING VARIANCE
           DO 380 M=1,ITWS       ! ALL THE COLUMNS
```

```
                N=I1-1+M
                R1=(FLOAT(IARR1(N,J-1))-RMEAN)*RFACT
                R2=(FLOAT(IARR1(N,J+ITWS-1))-RMEAN)*RFACT
                RARR1(1,M)=RARR1(1,M)-R1+R2       ! ADJUST SUM(X) FOR COLUMN M
                RARR1(2,M)=RARR1(2,M)-(R1*R1)+(R2*R2) ! ADJUST SUM(X**2)
                SUM1=SUM1+RARR1(1,M)              ! OVER ALL SUM(X)
                SUM2=SUM2+RARR1(2,M)              ! OVER ALL SUM(X**2)
380           CONTINUE
              RARR(L)=SQRT(SUM2/RN-(SUM1/RN)**2.0)*VART  ! VARIANCE FACTOR
C             WRITE(6,365)J,L,RARR(L),SUM1,SUM2,((RARR1(J1,J2),
C    +        J2=1,ITWS),J1=1,2)
            END IF
C
C           3.2.3 MOVE ALONG A ROW - EITHER TOWARDS THE RIGHT OR LEFT. WHEN
C                 TOWARDS THE RIGHT, THEN SHIFT THE COLUMN SUMS ONE POSITION
C           TO THE LEFT AND COMPUTE SUMS FOR THE NEW RIGHT HAND COLUMN - VICE
C           VERSA WHEN MOVING TOWARDS THE LEFT IN THE ROW.
C
            IL=ITWS                   ! THE LEADING EDGE COLUMN NUMBER
            IT=1                      ! THE TRAILING EDGE COLUMN NUMBER
            IF(I3.EQ.-1)THEN          ! OTHER WAY ROUND WHEN MOVING TOWARDS
              IL=1                    ! THE LEFT
              IT=ITWS
            END IF
C
            DO 500 I=I1+I3,I2,I3      ! SEARCH ARRAY COLUMN POSITION FOR TEMPLATE
                                      ! TOP-LEFT CORNER
              L=I+ITWS/2              ! COLUMN POSITION FOR WRITING THE VARIANCE.
              SUM1=SUM1-RARR1(1,IT)         ! SUBTRACT TRAILING COLUMN.
              SUM2=SUM2-RARR1(2,IT)
              DO 400 M=IT+I3,IL,I3          ! SHIFT COLUMN SUMS ONE POSITION
                RARR1(1,M-I3)=RARR1(1,M)    ! TOWARDS TRAILING THE EDGE
                RARR1(2,M-I3)=RARR1(2,M)
400           CONTINUE
C
              M=I           ! SEARCH ARRAY COLUMN CORRESPONDING TO LEADING
              IF(I3.EQ.1)M=I+ITWS-1         ! COLUMN
              RARR1(1,IL)=0.0               ! ZERO LEADING COLUMN SUM
              RARR1(2,IL)=0.0
              DO 420 N=J,J+ITWS-1           ! SUM THE ELEMENTS OF THE LEADING
                R1=(FLOAT(IARR1(M,N))-RMEAN)*RFACT        ! EDGE COLUMN.
                RARR1(1,IL)=RARR1(1,IL)+R1
                RARR1(2,IL)=RARR1(2,IL)+R1*R1
420           CONTINUE
              SUM1=SUM1+RARR1(1,IL)         ! ADD NEW COLUMN SUM TO OVER ALL SUM.
              SUM2=SUM2+RARR1(2,IL)
              RARR(L)=SQRT(SUM2/RN-(SUM1/RN)**2.0)*VART  ! VARIANCE FACTOR
C             WRITE(6,365)J,L,RARR(L),SUM1,SUM2,((RARR1(J1,J2),
C    +        J2=1,ITWS),J1=1,2)
500         CONTINUE                  ! END OF A ROW. STORE THE RESULTS .
C
C           WRITE(6,510)K,(RARR(I),I=1,ISWS)
C510        FORMAT(1X,I3,8F9.4,7(/,1X,3X,8F9.4))
C
            CALL ZMWLIN(IARR,K,IFMASK(3),IFMAPS(1025),7,  ! WRITE LINE OF
     +      IDECIM(3),NUMWPL(3),ESTADR(3))                ! VARIANCES TO IMAGE 3
            IT=I1                     ! PREPARE LOOP VARIABLES TO SHIFT THE
            I1=I2                     ! TEMPLATE WINDOW NOW IN THE OPPOSITE
            I2=IT                     ! DIRECTION
            I3=-1*I3
```

```
600     CONTINUE
C       WRITE(6,620)ITWSL,ITWLL,ITWSP,ITWLP,RM(1),VART,RM(2)
C620    FORMAT(1X/' ADVEC3 : TEMPLATE DIMENSIONS(L1,L2,P1,P2):',4I5,/,
C     +  ' TEMPLATE MEAN AND VARIANCE=',2F8.4,/,
C     +  ' SEARCH ARRAY MEAN=',F8.4,/)
        IFLAG=1
        GOTO 820
C
C       4.0 UPDATE WINDOW POSITION. IF PREVIOUS WINDOW WAS ABORTED THEN SET
C           IFLAG = 0, AND DO ANOTHER WINDOW WITHOUT GOING TO VECTOR EXTRACT-
C           ION STEP.
C
800     IFLAG=0
820     IWINDP=IWINDP+ISHIFT
        IF((IWINDP+ISWS-1).GT.ILPPR)THEN          ! END OF THIS LINE
          IWINDP=IFPPR            ! RESET WINDOW TO LEFTMOST PIXEL POSITION
          IWINDL=IWINDL+ISHIFT    ! SET FOR NEW LINE POSITION
        END IF
        IF(IFLAG.EQ.0.AND.(IWINDL+ISWS-1).LE.LIN2)THEN
          WRITE(5,840)
840       FORMAT('   WINDOW ABORT.')
          GOTO 50         ! WINDOW ABORTED. NO FFT. GO DO NEXT WINDOW.
        END IF
C
C
1000    RETURN
        END
```

```
C
C       SUBROUTINE : ADVEC4.FTN
C       THIS SUBROUTINE IS CALLED BY PROGRAM ADVECT WHICH PERFORMS AUTOMATIC
C       FEATURE TRACKING TO DERIVE SEA SURFACE ADVECTION VECTORS.
C
C       ADVEC4 CONVERTS THE TEMPLATE AND SEARCH WINDOW DATA STORED IN THE
C       TWO BUFFER IMAGES, IMAGE 1 AND IMAGE 2, BY ADVEC3 INTO ONE DIMENSIONAL
C       COMPLEX ARRAYS AND CALL SUBROUTINE FFT1 TO PERFORM FAST FOURIER
C       TRANSFORMS. THE RESULTANT 2-DIM FOURIER TRANSFORMS, F(U,V) AND G(U,V)
C       ARE STORED BACK IN THE IMAGE BUFFERS. THE ROUTINE THEN COMPUTES
C       R(U,V)=F'(U,V)G(U,V) , WHERE F'(U,V) IS THE COMPLEX CONJUGATE OF
C       THE 2-DIM TRANSFORM OF THE TEMPLATE ARRAY. NEXT THE INVERSE TRANSFORM
C       OF R(U,V) IS COMPUTED BY REPEATED CALLS TO FFT1. THE INVERSE TRANSFORM
C       WHICH IS THE CROSS COVARIANCE OF THE TEMPLATE AND SEARCH ARRAYS, IS
C       DIVIDED BY THE SQUARE ROOT OF THE TEMPLATE VARIANCE * SEARCH WINDOW
C       VARIANCE, READ FROM BUFFER IMAGE 3 - THIS GIVES THE REQUIRED
C       CORRELATION COEFFICIENT.
C       THE LINE/PIXEL COORDINATES OF THE VECTOR START POINT IS SAVED ALONG
C       WITH THE COORDINATES AND COEFFICIENTS OF THE 4 HIGHEST CORRELATION
C       POINTS FOUND WITHIN THAT PART OF THE SEARCH WINDOW WHERE THE TEMPLATE
C       FITS COMPLETELY INSIDE THE SEARCH WINDOW ( LIMITS DEFINED BY JL1
C       AND JL2)
C
C
C       WRITTEN BY J.J.AGENBAG  S.F.R.I.  APRIL 1990.  IN FORTRAN 77,
C       FOR A DEC LSI 11/23 BASED IMAGE PROCESSING SYSTEM. DIPIX ARIES II
C       SOFTWARE MODULES ARE USED FOR CERTAIN OPERATIONS.
C
C
      SUBROUTINE ADVEC4(ITWS,ISWS,ISWSL,ISWSP,QQQ,NVEC,IERR,
     +  NAMSUB,NAMMOD)
C
      IMPLICIT COMPLEX(A),BYTE(B),CHARACTER*1(C),INTEGER*4(E)
      DIMENSION ARR1(64),ARR2(64),IARR1(256),IARR2(256),IR(4),
     +  RARR(64),RCORR(3,4)
      COMMON IFMASK(3),IDECIM(3),IFMAPS(1536),NUMWPL(4),ESTADR(4)
      EQUIVALENCE (ARR1(1),IARR1(1)),(ARR2(1),IARR2(1),RARR(1)),
     +  (IR(1),AR)
C
      CHARACTER *7 NAMMOD, NAMSUB
      NAMSUB='ADVEC4 '
      RSWS=FLOAT(ISWS)                    ! SEARCH WINDOW SIZE
      IHSW=ISWS/2                         ! HALF SEARCH WINDOW.
      IHTW=ITWS/2                         ! HALF TEMPLATE SIZE.
      IVSL=ISWSL+IHSW           ! LINE AND PIXEL COORDINATES OF THE
      IVSP=ISWSP+IHSW           ! VECTOR START POINT - NEAR WINDOW CENTRE.
      JL1=1+IHTW                ! LIMITS FOR THE SECTION OF THE SEARCH
      JL2=ISWS-IHTW+1           ! WINDOW WHERE TEMPLATE FITS COMPLETELY
                                ! INSIDE THE WINDOW.
      DO 10 I=1,4               ! ZERO THE ARRAY USED TO STORE THE 4 HIGHEST
        RCORR(3,I)=0.0          ! CORRELATION COEFFICIENTS .
10    CONTINUE
C
C     WRITE(6,20)ITWS,ISWS,ISWSL,ISWSP,NVEC,IFMASK,IDECIM,
C    +  NUMWPL,ESTADR
C20   FORMAT(1X//' ADVEC4 1 : ITWS,ISWS,ISWSL,ISWSP=',4I6,/,
C    +  ' NVEC,IFMASK,IDECIM =',I3,3I5,3X,3I5,/,
C    +  ' NUMWPL,ESTADR =',4I5,3X,4I8,//,
C    +  ' ADVEC4 2 :  TEMPLATE COLUMN TRANSFORMS :',/)
```

```
C
C       1.0 PERFORM FOURIER TRANSFORM ON TEMPLATE ARRAY ( IMAGE 1)
C           NOTE : WHEN WRITING BACK THE COLUMNS IT IS NECESSARRY TO FIRST
C           READ THE FULL LINE CORRESPONDING TO THE COLUMN ELEMENTS SINCE
C           IT IS NOT POSSIBLE TO WRITE A PORTION ONLY OF A ROW.
C
        EFACT1=2*NUMWPL(1)
        EFACT2=ESTADR(1)-EFACT1
C
C       1.1 COLUMN TRANSFORMS
C
        NUMW=4                              ! NUMBER OF WORDS TO READ AT A TIME.
        DO 140 I=1,ISWS           ! THE COLUMNS
          IP=(I-1)*8
          DO 100 J=1,ISWS         ! THE ELEMENTS IN THE COLUMN
            EVADDR=EFACT2+EFACT1*J+IP
            CALL ZMRVMA(EVADDR,NUMW,QQQ,IERR,IR) ! READ ONE VALUE ( COMPLEX)
            IF(IERR.EQ.1)GOTO 80
             NAMMOD='ZMRVMA1'
             GOTO 1000                                ! EXIT WITH ERROR.
80          ARR1(J)=AR  ! STORE COLUMN ELEMENTS IN ARR()
100       CONTINUE
C         WRITE(6,105)I,(ARR1(J),J=1,ISWS)
C105      FORMAT(1X,I4,5(F8.3,F5.2,2X),50(/1X,5X,5(F8.3,F5.2,2X)))
          CALL FFT1(ARR1,ISWS)              ! DO FAST FOURIER ON COLUMN.
C         WRITE(6,107)((ARR1(J)/RSWS),J=1,ISWS)
C107      FORMAT(1X/16(1X,8F8.3,/))
          DO 120 J=1,ISWS                   ! DIVIDE BY ISWS AND WRITE BACK TO IMAG
            EVADDR=EFACT2+EFACT1*J          ! ADDRESS FOR FIRST BYTE IN THE ROW.
            CALL ZMRVMA(EVADDR,NUMWPL(1),QQQ,IERR,IARR2) ! READ ENTIRE ROW.
            IF(IERR.EQ.1)GOTO 110
             NAMMOD='ZMRVMA2'
             GOTO 1000                      ! EXIT WITH ERROR
110         ARR2(I)=ARR1(J)/RSWS
            CALL ZMWLIN(IARR2,J,IFMASK(1),IFMAPS(1),7,IDECIM(1),
     +      NUMWPL(1),ESTADR(1)) ! WRITE COLUMN ELEMENT AS PART OF THE
                                 ! ENTIRE ROW
C
C           IF(I.EQ.ISWS.AND.J.LE.3)WRITE(6,107)(ARR2(K),K=1,ISWS)
120       CONTINUE
140     CONTINUE                            ! NEXT COLUMN
C
C       1.2 ROW TRANSFORMS.
C
C       WRITE(6,145)
C145    FORMAT(1X//' ADVEC4 5 : TEMPLATE ROW TRANSFORMS :',/)
        NUMW=4*ISWS                         ! NUMBER OF WORDS TO READ AT A TIME
        DO 200 J=1,ISWS                     ! THE ROWS
          EVADDR=EFACT2+EFACT1*J
          CALL ZMRVMA(EVADDR,NUMW,QQQ,IERR,IARR1)  ! READ A ROW FROM IMAGE 1
C         WRITE(6,107)(ARR1(K),K=1,ISWS)
          IF(IERR.EQ.1)GOTO 160
           NAMMOD='ZMRVMA3'
           GOTO 1000                        ! EXIT WITH ERROR.
160       CALL FFT1(ARR1,ISWS)              ! DO FFT ON ROW
          CALL ZMWLIN(IARR1,J,IFMASK(1),IFMAPS(1),7,IDECIM(1),   ! WRITE BACK
     +    NUMWPL(1),ESTADR(1))                                   ! TO IMAGE 1
C         WRITE(6,107)(ARR1(K),K=1,ISWS)
200     CONTINUE
C
```

```
C       2.0 DO FOURIER TRANSFORM OF SEARCH ARRAY ( IMAGE 2).
C
        EFACT3=2*NUMWPL(2)
        EFACT4=ESTADR(2)-EFACT3
C
C       2.1 COLUMN TRANSFORMS.
C
        NUMW=4                      ! NUMBER OF WORDS TO READ AT A TIME.
C       WRITE(6,210)
C210    FORMAT(1X//' ADVEC4 8 : SEARCH ARRAY COLUMN TRANSFORMS :',/)
        DO 300 I=1,ISWS             ! THE COLUMNS
          IP=(I-1)*8
          DO 240 J=1,ISWS           ! ELEMENTS IN THE COLUMN
            EVADDR=EFACT4+EFACT3*J+IP
            CALL ZMRVMA(EVADDR,NUMW,QQQ,IERR,IR)  ! READ ONE VALUE (COMPLEX)
            IF(IERR.EQ.1)GOTO 220
             NAMMOD='ZMRVMA4'
             GOTO 1000              ! EXIT WITH ERROR.
220         ARR1(J)=AR              ! STORE IN ARR1() FOR FFT
240       CONTINUE
C         WRITE(6,107)(ARR1(K),K=1,ISWS)
          CALL FFT1(ARR1,ISWS)  ! PERFORM FFT
C         WRITE(6,107)((ARR1(J)/RSWS),J=1,ISWS)
          DO 260 J=1,ISWS           ! DIVIDE BY ISWS AND WRITE BACK TO IMAGE 2.
            EVADDR=EFACT4+EFACT3*J      ! ADDRESS OF 1ST BYTE IN THE ROW
            CALL ZMRVMA(EVADDR,NUMWPL(2),QQQ,IERR,IARR2)  ! READ ENTIRE ROW.
            IF(IERR.EQ.1)GOTO 250
             NAMMOD='ZMRVMA5'
             GOTO 1000              ! EXIT WITH ERROR.
250         ARR2(I)=ARR1(J)/RSWS
            CALL ZMWLIN(IARR2,J,IFMASK(2),IFMAPS(513),7,IDECIM(2),
     +      NUMWPL(2),ESTADR(2))  ! WRITE COLUMN ELEMENT AS PART OF INTIRE ROW
260       CONTINUE
300     CONTINUE
C       WRITE(6,305)
C305    FORMAT(1X//' ADVEC4 10 : SEARCH WINDOW ROW TRANSFORMS :'/)
C
C       2.2 ROW TRANSFORMS.   HAVING DONE EACH ROW TRANSFORM, READ THE
C           CORRESPONDING ROW FROM THE TRANSFORMED TEMPLATE (IN IMAGE 1),
C           COMPUTE THE CONJUGATE THERE-OF, MULTIPLY WITH THE SEARCH WINDOW
C           TRANSFORM. TAKE THE CONJUGATE OF THE PRODUCT AND PASS TO FFT1 FOR
C           THE INVERSE TRANSFORM. WRITE THE RESULT TO IMAGE 1.
C
        NUMW=4*ISWS     ! NUMBER OF WORDS TO READ AT A TIME
        DO 400 J=1,ISWS             ! THE ROWS.
          EVADDR=EFACT4+EFACT3*J        ! ADDRESS IMAGE 2
          EVAD=EFACT2+EFACT1*J          ! ADDRESS IMAGE 1
          CALL ZMRVMA(EVADDR,NUMW,QQQ,IERR,IARR1)  ! READ A ROW FROM IMAGE 2
C         WRITE(6,315)(ARR1(K),K=1,ISWS)
C315      FORMAT(' 11:',8F8.3,/,15(1X,3X,8F8.3,/))
          IF(IERR.EQ.1)GOTO 320
           NAMMOD='ZMRVMA5'
           GOTO 1000                    ! EXIT WITH ERROR.
320       CALL FFT1(ARR1,ISWS)          ! DO FFT ON ROW J
C         WRITE(6,325)(ARR1(K),K=1,ISWS)
C325      FORMAT(' 12:',8F8.3,/,15(1X,3X,8F8.3,/))
          CALL ZMRVMA(EVAD,NUMW,QQQ,IERR,IARR2)    ! READ THE ROW FROM IMAGE 1
C         WRITE(6,330)(ARR2(K),K=1,ISWS)
C330      FORMAT(' 13:',8F8.3,/,15(1X,3X,8F8.3,/))
          IF(IERR.EQ.1)GOTO 340
```

```
          NAMMOD='ZMRVMA6'
          GOTO 1000                          ! EXIT WITH ERROR
C
340       DO 360 I=1,ISWS          ! MULTIPLY THE ROW ELEMENTS
            ARR2(I)=CONJG(CONJG(ARR2(I))*ARR1(I))        ! (F'(U,V)G(U,V))'
360       CONTINUE
C         WRITE(6,107)(ARR2(I),I=1,ISWS)
          CALL FFT1(ARR2,ISWS)               ! DO THE INVERSE TRANSFORM ON ROW J
C         WRITE(6,107)(ARR2(I),I=1,ISWS)
          CALL ZMWLIN(IARR2,J,IFMASK(1),IFMAPS(1),7,IDECIM(1),
     +    NUMWPL(1),ESTADR(1))   ! WRITE BACK TO IMAGE 1
400     CONTINUE
C
C       3.0 COMPLETE THE INVERSE TRANSFORM OF F'(U,V)G(U,V) BY TRANSFORMING
C           THE COLUMNS OF IMAGE 1 ( WHICH NOW CONTAINS THE INVERSE ROW
C           TRANSFORMS). THEN EXTRACT THE REAL PART OF THE COMPLEX NUMBERS
C           (FROM THE INVERSE TRANSFORM) AND DIVIDE BY THE SQUARE ROOTS OF TH
C           WINDOW VARIANCES -FROM IMAGE3- TO OBTAIN THE CROSS CORRELATION
C           MATRIX. STORE THESE BACK IN IMAGE 3. WHILE DOING THIS KEEP TRACK
C           OF THE LARGEST VALUES.
C
C           NOTE !! : IN ORDER TO GET THE COVARIANCE COEFFICIENTS ( IE. THE
C                     OPUTUT FROM THE INVERSE TRANSFORM) IN TO THE CORRECT
C           POSITIONS THE RESULTS HAVE TO BE SHUFFLED. THE FIRST HALF OF THE
C           SEARCH ARRAY GO TO THE END AND THE LAST HALF TO THE FRONT. SAME
C           FOR BOTH COLUMNS AND ROWS.
C
          EFACT3=2*NUMWPL(3)
          EFACT4=ESTADR(3)-EFACT3
          NUMW=4                             ! NUMBER OF WORDS TO READ AT A TIME.
          NUMW1=ISWS*2
          IC1=IHSW+1                ! SET VARIABLES TO DO SECOND  HALF OF
          IC2=ISWS                  ! COLUMNS FIRST
          IC=0
C         WRITE(6,405)
C405      FORMAT(1X//' FINAL INVERSE TRANSFORM OF COLUMNS :- REAL PARTS',
C     +   ' PRINTED IN ROWS.',/,' CORRESPONDING VARIANCES FROM IMAGE 3',
C     +   ' BELOW THAT( COLUMNS AS ROWS) :',/)
410       DO 500 I=IC1,IC2                          ! THE COLUMNS
            IC=IC+1
            IP=(I-1)*8
            DO 440 J=1,ISWS                 ! ELEMENTS IN COLUMN I
              EVADDR=EFACT2+EFACT1*J+IP
              CALL ZMRVMA(EVADDR,NUMW,QQQ,IERR,IR)
              IF(IERR.EQ.1)GOTO 420
               NAMMOD='ZMRVMA7'
               GOTO 1000                     ! EXIT WITH ERROR
420           ARR1(J)=AR                    ! STORE IN ARR1() FOR FFT
440         CONTINUE
C           WRITE(6,445)(ARR1(K),K=1,ISWS)
C445        FORMAT(' 14:',8F8.3,/,15(1X,3X,8F8.3,/))
            CALL FFT1(ARR1,ISWS)            ! TRANSFORM COLUMN J
C           WRITE(6,446)I,(REAL(ARR1(K)),K=1,ISWS)
C446        FORMAT(1X,I3,8F9.4,/,(1X,3X,8F9.4,/))
            K=0
            J1=1                    ! SET VARIABLES TO DO FIRST HALF
            J2=IHSW                 ! OF SEARCH ARRAY ROWS.
            JE=IHSW
448         DO 460 J=J1,J2          ! COMPUTE CROSS CORRELATION ON COLUMN ELEMENTS
              JE=JE+1
```

```
            K=K+1
            EVADDR=EFACT4+EFACT3*J    ! START ADDRESS FOR THIS ROW
            CALL ZMRVMA(EVADDR,NUMW1,QQQ,IERR,IARR2)  ! READ A LINE OF VARIANCE
C           VARR(K)=RARR(IC)
            RCOV=REAL(ARR1(JE))
            IF(IERR.EQ.1)GOTO 450
             NAMMOD='ZMRVMA8'
             GOTO 1000                    ! EXIT WITH ERROR
450         IF(ABS(RARR(IC)).LT.0.001)RARR(IC)=.001
            RCOR=RCOV/(RSWS*RARR(IC))
C           IF(K.EQ.8)THEN
C             WRITE(6,455)IC,RCOV,VARR
C455          FORMAT(' C',I2,3X,8F9.4,/,1X,6X,8F9.4)
C             K=0
C           END IF
C           CALL ZMWLIN(IARR2,J,IFMASK(3),IFMAPS(1025),7,IDECIM(3),
C     +     NUMWPL(3),ESTADR(3))          ! WRITE ROW BACK TO IMAGE 3.
C
            IF(RCOR.GT.RCORR(3,4))THEN
              IF(IC.LT.JL1.OR.IC.GT.JL2)GOTO 460
              IF(K.LT.JL1.OR.K.GT.JL2)GOTO 460
              DO 456 IN=1,4
                IF(RCOR.GT.RCORR(3,IN))GOTO 457
456           CONTINUE
457           DO 459 JN=3,IN,-1
                DO 458 KN=1,3
                  RCORR(KN,JN+1)=RCORR(KN,JN)
458             CONTINUE
459           CONTINUE
              RCORR(1,IN)=FLOAT(K+ISWSL-1)
              RCORR(2,IN)=FLOAT(IC+ISWSP-1)
              RCORR(3,IN)=RCOR
            END IF
460       CONTINUE
C         WRITE(6,470)
C470      FORMAT(' ')
          IF(J1.EQ.1)THEN                 ! SET VARIABLES TO DO SECOND HALF
            J1=IHSW+1                     ! OF SEARCH ARRAY ROWS.
            J2=ISWS
            JE=0
            GOTO 448
          END IF
500     CONTINUE
        IF(IC2.EQ.ISWS)THEN
          IC1=1                   ! SET VARIABLES TO DO FIRST HALF OF COLUMNS.
          IC2=IHSW
          GOTO 410
        END IF
C
C       4.0 ZERO THE TEMPLATE STORAGE IMAGE 1 AND SAVE THE VECTOR DATA IN
C           FILE XXXVEC.DAT (=UNIT #4)
C
800     DO 820 I=1,256
          IARR1(I)=0
820     CONTINUE
        DO 840 I=1,ISWS
          CALL ZMWLIN(IARR1,I,IFMASK(1),IFMAPS(1),7,IDECIM(1),
     +    NUMWPL(1),ESTADR(1))
840     CONTINUE
C
```

```
      WRITE(6,900)IVSL,IVSP,((INT(RCORR(1,I)),INT(RCORR(2,I)),
     +  RCORR(3,I)),I=1,4)
900   FORMAT(2I5,4(1X,2I4,F8.3))
C
C
C
1000  RETURN
      END
```

```
C
C       SUBROUTINE FFT1.FTN
C       SUBROUTINE TO PERFORM A ONE DIMENSIONAL FAST FOURIER TRANSFORM ON
C       THE COMPLEX ARRAY 'ARRAY'. THE RESULT IS RETURNED VIA THE SAME ARRAY.
C
C       ALGORITHM FROM . 'WORKSHOP ON IMAGE PROCESSING (PART 1).
C       DEPT. OF COMPUTER SCIENCE , UNIV. OF NATAL. 23-24 JUL 1984.
C       EDITOR : PROF. A.G.SARTORI-ANGUS.
C
C       J.J.AGENBAG  S.F.R.I.  AFRIL 1990 .
C       IN FORTRAN 77 FOR A DEC LSI 11/23
C
C
        SUBROUTINE FFT1(ARRAY,N)
        IMPLICIT COMPLEX(A)
        DIMENSION ARRAY(N)
C
        RPI=4.0D0*DATAN(1.0D0)              ! PI
        LOG2N=INT(ALOG(FLOAT(N))/ALOG(2.0)+0.5)
C
C       1.0 RE-ORDER THE ELEMENTS OF 'ARRAY' FOR FAST FOURIER TRANSFORM.
C
        J=1
        DO 100 I=1,N-1
          IF(I.LT.J)THEN
            AT=ARRAY(J)
            ARRAY(J)=ARRAY(I)
            ARRAY(I)=AT
          END IF
          K=N/2
60        IF(K.LT.J)THEN
            J=J-K
            K=K/2
            GOTO 60
          END IF
          J=J+K
100     CONTINUE
C
C       FOURIER TRANSFORM
C
        RLE=1.0
        DO 200 L=1,LOG2N
          RLE1=RLE
          RLE=RLE+RLE
          AU=(1.0,0.0)
          AW=CMPLX(COS(RPI/RLE1),-SIN(RPI/RLE1))
          LE=INT(RLE)
          LE1=INT(RLE1)
          DO 180 J=1,LE1
            DO 160 I=J,N,LE
              IP=I+LE1
              AX=ARRAY(I)
              AT=ARRAY(IP)*AU
              ARRAY(IP)=AX-AT
              ARRAY(I)=AX+AT
160         CONTINUE
            AU=AU*AW
180       CONTINUE
200     CONTINUE
```

```
C
C
      RETURN
      END
```

```
;
;  TASK BUILD COMMAND FILE FOR 'ADVECT' - DERIVE SEA SURFACE ADVECTION
;  VECTORS THROUGH AUTOMATED FEATURE TRACKING.
;
   ADVECT/FP/CP=ADVECT.ODL/MP
   COMMON=DPICOM:RW:5
   UNITS=6
   ASG=IM:1,DD1:2,SY:3,SY:4,TI:5,TT1:6
;
;  GET REMAINING INFORMATION FROM ADVECT.ODL
;
```

```
;
; OVERLAY DESCRIPTOR FILE FOR PROGRAM 'ADVECT' - DERIVE SEA SURFACE
; ADVECTION VECTORS THROUGH AUTOMATED FEATURE TRACKING.
;
        .ROOT ADVECT-*(A1,A2,A3,A4)
;
A1:     .FCTR ADVEC1-LB1
A2:     .FCTR ADVEC2-LB2
A3:     .FCTR ADVEC3-LB2
A4:     .FCTR ADVEC4-FFT1-LB2
;
LB1:    .FCTR DR:[100,300]COMMON/LB
LB2:    .FCTR DR:[100,300]VMALIB/LB
;
.END
;
;       .-------------------------------------------------------
```

APPENDIX D

Listing of program 'AUTOTR'

```
C
C       PROGRAM AUTOTR.FTN
C       THIS PROGRAM PERFORMS TEMPLATE MATCHING BETWEEN TWO IMAGES, USING
C       THE CORRELATION COEFFICIENT DESCRIBED BY SVEDLOV ET. AL.(IEEE TRAN-
C       SACTIONS ON AEROSPACE AND ELECTRONICS, VOL.AES-14,NO.1,JAN.1978) -
C       EQUATION NO.1 IN TABLE 1. PAGE 143 - AS THE MEASURE OF MATCH.
C       THE PROGRAM WAS DEVELOPED TO DERIVE SEA SURFACE ADVECTION VECTORS
C       IN A MANNER SIMILAR TO THAT BY EMERY ET.AL. 1986, J.G.R. P12865-
C       12878, VOL.91,NO.C11
C       THE PROCEDURE REQUIRES TWO IMAGES SAY 12 OR 24 HOURS APART, AND
C       EITHER REGISTERED TO EACH OTHER OR TO A COMMON MAP PROJECTION.
C       THE IMAGES SHOULD BE GRADIENT IMAGES AND THE LAND AND CLOUD MASKED
C       BY PIXEL COUNTS OF 255.
C       THE USER THEN SPECIFY THE FILE NAMES AND SECTION OF THE IMAGE TO
C       BE PROCESSED. ALSO THE TEMPLATE WINDOW SIZE N1 (N1 LINES X N1 PIXELS
C       IN THE FIRST IMAGE) AND A SEARCH WINDOW SIZE N2 ( N2 X N2 IN 2'ND
C       IMAGE ). N1<N2 . N1 AND N2 MUST BE ODD  AND NOT BIGGER THAN 64.
C       THE PROGRAM WILL ATTEMPT TO DERIVE THE START AND END POINTS FOR AN
C       ADVECTION VECTOR IN THE WINDOWS ( STARTING AT THE TOP LEFT CORNER
C       OF THE SPECIFIED IMAGE SECTION) USING THE PRINCIPLE OF MAXIMUM
C       CORRELATION COEFFICIENT (MCC). IF THE MCC IS LESS THAN 0.4 OR
C       IF THE SPECIFIED WINDOWS CONTAIN MORE THAN 1% CLOUD OR LAND THEN NO
C       VECTOR WILL BE CALCULATED. OTHERWISE THE START LINE/PIXEL AND END
C       LINE/PIXEL NUMBERS ARE RECORDED ALONG WITH THE MCC AND THE TEMPLATE
C       VARIANCE IN A FILE NAMED XXXVEC.DAT, WHERE XXX= THE AREA CODE FOR
C       THE SPECIFIED IMAGES.
C       WRITING TO THE FILE IS DONE BY : WRITE(4,*)L1,P1,L2,P2,RMAXCC,VAR
C
C       THE USER ALSO SPECIFY A WINDOW SHIFT, WHICH IS THE NUMBER OF LINES/
C       PIXELS BY WHICH THE WINDOWS ARE SHIFTED ALONG FOR SUCCESSIVE VECTOR
C       DETERMINATIONS - IE. IT DETERMINES POTENTIAL VECTOR DENSITY.
C
C       A TOTAL OF 4 SUBROUTINES ARE USED :
C        1. AUTOT1 : GET FILE NAMES, IMAGE SECTION TO PROCESS, WINDOW SIZES
C           AND SHIFT FACTOR. IT ALSO CREATES THE STORAGE FILE XXXVEC.DAT
C
C        2. AUTOT2 : USES THE INFORMATION PROVIDED BY AUTOT1 TO CREATE A
C           16-BIT , NONDECIMATED IMAGE FOR THE SPECIFIED IMAGE AREA -
C           IF NOT ENOUGH VMA AVAILABLE TO LOAD THE ENTIRE AREA, THEN AS
C           LARGE A NUMBER OF LINES OF SPECIFIED NO. OF PIXELS AS POSSIBLE
C           WILL BE LOADED AND FURTHER VISITS TO AUTOT2 WILL BE NECESSARY TO
C           PROCESS THE ENTIRE AREA. IT LOADS THE SPECIFIED TO FEATURES IN
C           TO THE CREATED IMAGE.
C
C        3. AUTOT3 : CALCULATES SUCCESSIVE POSITIONS OF THE WINDOWS AND READ
C           FROM VMA THE DATA, CONVERT TO INTEGER, CALCULATE THE TEMPLATE
C           VARIANCE AND CALLS CRCOEF TO DO THE TEMPLATE MATCHING. IT
C           RECORDS THE RESULTS IN FILE XXXVEC.DAT
C
C
C       WRITTEN BY J.J.AGENEAG  S.F.R.I.  JUNE 1990. IN FORTRAN 77.
C       FOR A DEC LSI 11/23 BASED IMAGE PROCESSING SYSTEM USING DIPIX
C       ARIES II SOFTWARE MODULES FOR CERTAIN OPERATIONS.
C
        PROGRAM AUTOTR.
C
        IMPLICIT COMPLEX(A),BYTE(B),CHARACTER*1(C),INTEGER*4(E)
        DIMENSION BAR(3),BFINAM(9,2)
        CHARACTER*7 NAMMOD, NAMSUB , VECTFN*10
```

```
C
      ICYC=1              ! FIRST VISIT TO AUTOT2
      NVEC=0              ! COUNTER FOR NUMBER OF DERIVED VECTORS.
C
C     1.0 CALL AUTOT1 TO INPUT DATA .
C
      CALL AUTOT1(QQQ,BAR,BFINAM,IFLPR,ILLPR,IFPPR,ILPPR,
     +VECTFN,ITWS,ISWS,ISHIFT,IERR,NAMMOD,NAMSUB)
      IF(IERR.NE.1)GOTO 80     ! EXIT WITH ERROR.
C
C     2.0 CALL AUTOT2 TO CREATE THE IMAGE OR LOAD ANOTHER IMAGE SECTION.
C
20    CALL AUTOT2(ICYC,BAR,BFINAM,IFLPR,ILLPR,IFPPR,ILPPR,
     +QQQ,LIN1,LIN2,ESTADR,ENWPL,IERR,NAMMOD,NAMSUB,
     +RRR,IWINDL,IMGNUM)
      IF(IERR.NE.1)GOTO 80
C
C     3.0 CALL AUTOT3 TO READ DATA FROM VMA AND DO THE FEATURE TRACKING.
C
C40   WRITE(5,50)ISWS,ITWS,ISHIFT,IFPPR,ILLPR,ILPPR,LIN1,LIN2,
C    +ESTADR,ENWPL,IFOUR,ISWSL,ISWSP,IERR,IWINDL,IWINDP
50    FORMAT(1X/' AUTOT3 :',8I8,/,1X,8I8)
C
      CALL AUTOT3(ISWS,ITWS,ISHIFT,IFPPR,ILLPR,ILPPR,LIN1,LIN2,
     +ESTADR,ENWPL,QQQ,NUMV,ISWSL,ISWSP,IERR,NAMMOD,NAMSUB,
     +RRR,IWINDL,IWINDP)
C     WRITE(5,50)ISWS,ITWS,ISHIFT,IFPPR,ILLPR,ILPPR,LIN1,LIN2,
C    +ESTADR,ENWPL,IFOUR,ISWSL,ISWSP,IERR,IWINDL,IWINDP
      IF(IERR.NE.1)GOTO 80     ! EXIT WITH ERROR.
      NVEC=NVEC+NUMV
      IF(LIN2.LT.ILLPR)THEN
         ICYC=ICYC+1
         GOTO 20              ! GO LOAD NEXT IMAGE SECTION.
       ELSE
         GOTO 100             ! END
      END IF
C
C     4.0 END
C
80    WRITE(5,90)IERR,NAMSUB,NAMMOD
90    FORMAT(1X//' EXITTING WITH ERROR',I4,'  IN SBROUTINE ',A7,/,
     +' MODULE ',A7)
C
100   WRITE(5,120)BFINAM,VECTFN,NVEC
120   FORMAT(1X//' IMAGE FEATURES ',9A1,' AND ',9A1,/,
     +' NUMBER OF VECTORS SAVED IN FILE ',A10,' =',I6)
      CLOSE(4)
      CALL EXIT
      END
```

```
C
C       SUBROUTINE AUTOT1.FTN
C       A SUBROUTINE CALLED BY PROGRAM AUTOTR WHICH PERFORMS AUTOMATIC
C       FEATURE TRACKING TO DERIVE SURFACE ADVECTION VECTORS.
C       AUTOT1 OBTAINS FILE NAMES, AREA TO PROCESS, WINDOW SIZES AND SHIFT.
C       A FILES-11 DATA FILE IS CREATED : XXXVEC.DAT TO STORE THE VECTOR
C       START END END LINES AND PIXELS ( XXX= AREA CODE FOR IMAGE)
C
C       WRITTEN BY J.J.AGENBAG  S.F.R.I.  JUNE 1990. IN FORTRAN 77.
C       FOR A DEC LSI 11/23 BASED IMAGE PROCESSING SYSTEM USING DIPIX
C       ARIES II SOFTWARE MODULES FOR CERTAIN OPERATIONS.
C
        SUBROUTINE AUTOT1(QQQ,BAR,BFINAM,IFLFR,ILLFR,IFPFR,ILPFR,
     +  VECTFN,ITWS,ISWS,ISHIFT,IERR,NAMMOD,NAMSUB)
C
        IMPLICIT COMPLEX(A),BYTE(B),CHARACTER*1(C),INTEGER*4(E)
        DIMENSION BAR(3),BFINAM(9,2),BFOUND(9),IBH(2),IBL(2)
        CHARACTER NAMMOD*7,NAMSUB*7,VECTFN*10
C
        PARAMETER (LUNIMG=1)
C
C
        NAMSUB='AUTOT1 '
        IERR=1
        VECTFN='***VEC.DAT'
C
C       1.0 GET IMAGE FILE NAMES. CHECK WHETHER THEY EXIST. GET DIMENSIONS.
C
10      WRITE(5,20)
20      FORMAT(1X/' ENTER 3-CHAR. AREA CODE FOR THE IMAGE TO BE ',
     +  'PROCESSED :',$)
        READ(5,40)VECTFN(1:3)
40      FORMAT(A3)
        DO 60 I=1,3                ! CONSTRUCT FILE NAME.
          WRITE(CHAR,50)VECTFN(I:I)
50        FORMAT(A1)
          READ(CHAR,50)BAR(I)
          BFINAM(I,1)=BAR(I)
          BFINAM(I,2)=BAR(I)
60      CONTINUE
        DO 80 I=4,5
          BFINAM(I,1)='F'
          BFINAM(I,2)='F'
80      CONTINUE
        DO 100 I=6,9
          BFINAM(I,1)='*'
          BFINAM(I,2)='*'
100     CONTINUE
        CALL ZFSEEK(BFINAM(1,1),1,QQQ,IERR,ISIZE,IBH(1),IBL(1),BFOUND)
        IF(IERR.EQ.1)GOTO 140    ! FILES WITH THIS AREA CODE EXIST.
        IF(IERR.EQ.-261)THEN     ! NO FILES WITH THIS CODE.
           WRITE(5,120)BAR
120        FORMAT(1X//' NO FILES WITH AREA CODE ',3A1,' FOUND.',
     +     ' RESPECIFY PLEASE.')
           GOTO 10
          ELSE
           NAMMOD='ZFSEEK1'      ! SOME OTHER ERROR.
           GOTO 1000
        END IF
```

```
C
C       AREA CODE EXIST. GET DIMENSIONS.
C
140     CALL ZHRDIM(BFINAM(1,1),LUNIMG,QQQ,IERR,NLIN,NPIX,IFP,IFL,ILP,
     +  ILL)
        IF(IERR.EQ.1)GOTO 160
        NAMMOD='ZHRDIM '
        GOTO 1000
C
C       INPUT FILE ID'S. CHECK.
C
160     WRITE(5,170)
170     FORMAT(1X/' ENTER THE FILE ID S FOR THE TWO FEATURE FILES',/,
     +  ' ON WHICH FEATURE TRACKING IS TO BE PERFORMED. THE FIRST ',/,
     +  ' WILL BE THE TEMPLATE IE. THE FIRST IMAGE IN TIME.')
        DO 300 I=1,2
180       WRITE(5,190)I
190       FORMAT(1X/' FOR FEATURE NO.',I2,' ENTER 4-CHAR FILE ID :',$)
          READ(5,200)(BFINAM(J,I),J=6,9)
200       FORMAT(4A1)
          CALL ZFSEEK(BFINAM(1,I),1,QQQ,IERR,ISIZE,IBH(I),IBL(I),BFOUND)
          IF(IERR.EQ.1)GOTO 300     ! FILE EXIST.
          IF(IERR.EQ.-261)THEN      ! FILE DO NOT EXIST.
             WRITE(5,240)(BFINAM(J,I),J=1,9)
240          FORMAT(1X/' FILE ',9A1,' DO NOT EXIST. RESPECIFY FLSE.')
             GOTO 180
            ELSE
             NAMMOD='ZFSEEK2'       ! SOME OTHER ERROR.
             GOTO 1000
          END IF
300     CONTINUE
C
C       2.0 GET AREA TO BE PROCESSED.
C
        WRITE(5,320)IFL,ILL,IFP,ILP
320     FORMAT(1X//' FILE DIMENSIONS ARE :',/,
     +  ' FIRST LINE =',I5,'  LAST LINE =',I5,/,
     +  ' FIRST PIXEL=',I5,'  LAST PIXEL=',I5)
330     WRITE(5,340)
340     FORMAT(1X/' ENTER FIRST AND LAST LINE TO PROCESS:',$)
        READ(5,*)IFLPR,ILLPR
        IF(IFLPR.LT.IFL.OR.ILLPR.GT.ILL.OR.ILLPR.LT.IFLPR) GOTO 330
350     WRITE(5,360)
360     FORMAT(1X/' ENTER FIRST AND LAST PIXEL TO PROCESS :',$)
        READ(5,*)IFPPR,ILPPR
        IF(IFPPR.LT.IFP.OR.ILPPR.GT.ILP.OR.ILPPR.LT.IFPPR)GOTO 350
        NUMPIX=ILPPR-IFPPR+1
        NPIX=2*INT(FLOAT(NUMPIX)/2.0)
        IF(NPIX.NE.NUMPIX)ILPPR=ILPPR+1
        IF(ILPPR.GT.ILP)ILPPR=ILPPR-2           ! PIXELS MUST BE EVEN.
        NUMPIX=ILPPR-IFPPR+1
C
C       3.0 ENTER THE WINDOW SIZES AND WINDOW SHIFT DISTANCE.
C
        WRITE(5,380)
380     FORMAT(1X//' TEMPLATE AND SEARCH WINDOWS WILL BE N1XN1 AND',/,
     +  ' N2XN2 PIXELS RESPECTIVELY. N1 MUST BE LESS THAN N2 AND ',/,
     +  ' BOTH MUST BE ODD AND NO LARGER THAN 64.')
390     WRITE(5,400)
400     FORMAT(1X/' ENTER N1 AND N2 :',$)
```

```
      READ(5,*)ITWS,ISWS
      ITW=2*INT(FLOAT(ITWS)/2.0)
      ISW=2*INT(FLOAT(ISWS)/2.0)
      IF(ITWS.GE.ISWS.OR.ITWS.EQ.ITW.OR.ISWS.EQ.ISW)GOTO 390
      IF(ITWS.GT.64.OR.ISWS.GT.64)GOTO 390
      IWINDL=0
410   WRITE(5,420)
420   FORMAT(1X//' ENTER THE WINDOW SHIFT DISTANCE FOR SUCCESSIVE',/,
     +' VECTOR CALCULATIONS ( SHOULD BE >=1) :',$)
      READ(5,*)ISHIFT
      IF(ISHIFT.LT.1)GOTO 410
C
C     4.0 CREATE THE STORAGE FILE FOR VECTOR COORDINATES .
C
      OPEN(4,FILE=VECTFN,STATUS='NEW')
C
C     6.0 RETURN
C
1000  RETURN
      END
```

```
C
C       SUBROUTINE AUTOT2.FTN
C       A SUBROUTINE CALLED BY PROGRAM AUTOTR WHICH PERFORMS AUTOMATIC
C       FEATURE TRACKING TO DERIVE SURFACE ADVECTION VECTORS.
C       AUTOT2 USES THE INFORMATION SUPPLIED BY AUTOT1 TO CREATE A 16-BIT,
C       NON-DECIMATED IMAGE AND LOAD THE SPECIFIED AREA OF THE TWO FEATURES.
C       IF NOT ENOUGH VMA AVAILABLE , AUTOT2 WILL LOAD AS LARGE A  NUMBER
C       OF LINES ( OF SPECIFIED NUMBER OF PIXELS) AS WILL FIT INTO VMA.
C       THE ROUTINE WILL THEN BE VISITED AGAIN TO PROCESS THE REMAINING
C       PART(S) OF THE IMAGE.
C
C       WRITTEN BY J.J.AGENBAG  S.F.R.I.  JUNE 1990. IN FORTRAN 77.
C       FOR A DEC LSI 11/23 BASED IMAGE PROCESSING SYSTEM USING DIPIX
C       ARIES II SOFTWARE MODULES FOR CERTAIN OPERATIONS.
C
        SUBROUTINE AUTOT2(ICYC,BAR,BFINAM,IFLPR,ILLPR,IFPPR,ILPPR,
     +  QQQ,LIN1,LIN2,ESTADR,ENWPL,IERR,NAMMOD,
     +  NAMSUB,RRR,IWINDL,IMGNUM)
C
        IMPLICIT COMPLEX(A),BYTE(B),CHARACTER*1(C),INTEGER*4(E)
        DIMENSION BAR(3),BFINAM(9,2),BTNAME(13),BIXD(2),BNFEAT(2),
     +  BFDPTH(2),BDUM(2),BNTFIL(2),BNTHMS(2)
        CHARACTER NAMMOD*7,NAMSUB*7
C
        DATA IDEPTH,IFEAT,IFD,NTFIL,NTHMS/15,2,8,0,0/
        DATA BTNAME/'I','N','I','S','F','E','V','M','A','.','S','F','F'/
        FARAMETER (LUNDD=2)
C
        EQUIVALENCE (BIXD(1),IDEPTH),(BNFEAT(1),IFEAT),(BDUM(1),IFD),
     +  (BNTFIL(1),NTFIL),(BNTHMS(1),NTHMS)
C
        BFDPTH(1)=BDUM(1)
        BFDPTH(2)=BDUM(1)
        NAMSUB='AUTOT2 '
        IERR=1
        IF(ICYC.NE.1)GOTO 110
C
C       1.0 ATTACH VIDEO SUBSYSTEM AND PERFORM IMAGE CREATION.
C
        CALL ZIGVID(LUNDD,QQQ,IERR)
        IF(IERR.EQ.1)GOTO 100
        NAMMOD='ZIGVID '
        GOTO 1000                   ! EXIT WITH ERROR.
C
100     CALL ZITOPL(BTNAME,QQQ,IERR)
        IF(IERR.EQ.1)GOTO 120
        NAMMOD='ZITOPL '
        GOTO 1000                   ! EXIT WITH ERROR
C
110     CALL ZIDELE(IMGNUM,'N',QQQ,IERR)
        IF(IERR.EQ.1)GOTO 120
        NAMMOD='ZIDELE '
        GOTO 1000
C
120     CALL ZIFREE(QQQ,IERR,EVMTOT,EVMCON)      ! HOW MUCH FREE VMA ?
        IF(IERR.EQ.1)GOTO 140
        NAMMOD='ZIFREE '
        GOTO 1000                   ! EXIT WITH ERROR.
C
```

```
C       1.1 CALCULATE HOW MANY LINES MAY BE LOADED AS 16-BIT NONDECIMATED.
C
140     RNL=FLOAT(EVMCON-152)/FLOAT(ILPPR-IFPPR+1)
        IF(RNL.GT.10000)RNL=10000.0
        NUMBLN=INT(RNL)
        IF(ICYC.EQ.1)THEN                    ! FIRST VISIT TO AUTOT2.
           LIN1=IFLPR                ! FIRST LINE TO LOAD.
           LIN2=ILLPR                ! LAST   ,,  ,,  ,,
         ELSE
           LIN1=IWINDL             ! NEXT LINE TO PROCESS. FROM ADVEC3
           LIN2=ILLPR
        END IF
        IF(NUMBLN.LT.(LIN2-LIN1+1))LIN2=LIN1+NUMBLN-1
        NUMLIN=LIN2-LIN1+1
C
C       1.2 CREATE THE IMAGE AND LOAD THE FEATURES.
C
        RDEC=1.0
        NUMPIX=ILPPR-IFPPR+1
        ENWPL=NUMPIX
        CALL ZICREA(BAR,BIXD(1),LIN1,IFPPR,NUMLIN,NUMPIX,BNFEAT(1),
     +  BFDPTH,BNTFIL(1),BNTHMS(1),'P','N','N',1.0,QQQ,IERR,
     +  IMGNUM,ESTADR,RRR,RDEC)
        IF(IERR.EQ.1)GOTO 160
        NAMMOD='ZICREA '
        GOTO 1000                    ! EXIT WITH ERROR.
C
160     ISATUR=255
        ICUTOF=0
        DO 200 I=1,2
          CALL ZIFEAT(IMGNUM,I,'C','F',QQQ,IERR,RRR,ISATUR,
     +    ICUTOF,BFINAM(1,I))
          IF(IERR.EQ.1)GOTO 200
          NAMMOD='ZIFEAT '
          GOTO 1000                  ! EXIT WITH ERROR.
200     CONTINUE
C
C       2.0 RETURN
C
1000    RETURN
        END
```

```
C
C       SUBROUTINE AUTOT3.FTN
C       A SUBROUTINE CALLED BY PROGRAM AUTOTR WHICH PERFORMS AUTOMATIC
C       FEATURE TRACKING TO DERIVE SURFACE ADVECTION VECTORS.
C       AUTOT3 CALCULATES THE LOCATIONS OF SUCCESSIVE TEMPLATE AND SEARCH
C       WINDOWS IN VMA AND READ THE DATA. CONVERT INTO INTEGER ARRAYS.
C       THE SEARCH WINDOW DATA ARE STORED IN ARRAY ISARR() AND THE TEMPLATE
C       DATA IN ARRAY ITARR().
C
C       CLOUD AND LAND MUST BE SET TO 255 IN THE FEATURES. ADVEC3 REJECTS
C       ANY WINDOW POSITION WHERE MORE THAN 1% OF PIXELS IN EITHER
C       THE TEMPLATE OR SEARCH IMAGE CONTAIN LAND OR CLOUD.
C       THE MAXIMUM WINDOW SIZE HANDLED BY THIS ROUTINE IS 64 X 64 (SEARCH
C       IMAGE) AND 32X32 (TEMPLATE IMAGE)
C
C       WRITTEN BY J.J.AGENBAG  S.F.R I.  JUNE 1990. IN FORTRAN 77.
C       FOR A DEC LSI 11/23 BASED IMAGE PROCESSING SYSTEM USING DIPIX
C       ARIES II SOFTWARE MODULES FOR CERTAIN OPERATIONS.
C
C
      SUBROUTINE AUTOT3(ISWS,ITWS,ISHIFT,IFPPR,ILLPR,ILPPR,
     +  LIN1,LIN2,ESTADR,ENWPL,QQQ,NUMV,ISWSL,ISWSP,IERR,NAMMOD,
     +  NAMSUB,RRR,IWINDL,IWINDP)
C
      IMPLICIT COMPLEX(A),BYTE(B),CHARACTER*1(C),INTEGER*4(E)
      DIMENSION IRVMA(64),BRVMA(2,64),BCON(2)
      COMMON/WINCOM/ISARR(64,64),ITARR(32,32)
      CHARACTER*7 NAMMOD,NAMSUB
      EQUIVALENCE (BRVMA(1,1),IRVMA(1)),(BCON(1),ICON)
C
      NUMV=0            ! COUNTER FOR EXTRACTED VECTORS.
      IERR=1
      RN2=(FLOAT(ITWS))**2.0          ! N1 X N1  FOR TEMPLATE VARIANCE
      RNF=RN2*(RN2-1)                 ! A CONSTANT IN TEMPLATE VARIANCE
C
      NAMSUB='AUTOT3 '
      ICON=0
C
C     1.0 SET TOP LEFT CORNER OF FIRST WINDOW TO TOP LEFT CORNER OF DISPLAY
C
      IF(ISWS.GT.64)THEN
        WRITE(5,20)NAMSUB
20      FORMAT(1X//1X,A7,' : ERROR 0 , SEARCH WINDOW TOO LARGE.',/)
        IERR=0
        NAMMOD='SEARCHW'
        GOTO 1000
      END IF
C
      IWINDL=LIN1
      IWINDP=IFPPR
      ISWL=IWINDL+ISWS/2        ! START LINE NO. FOR VECTOR.
      ITWLIM=INT(FLOAT(ITWS*ITWS)*.01+0.5)     ! LAND/CLOUD LIMIT TEMPLATE
      ISWLIM=INT(FLOAT(ISWS*ISWS)*.01+0.5)   ! SEARCH WINDOW LIMIT LAND/CLOU
C
C     2.0 WINDOW DIMENSIONS AND ADRESSES.
C
      E2WPL=2*ENWPL
      EVA0=ESTADR-LIN1*E2WPL-IFPPR*2
100   ISWLL=IWINDL+ISWS-1       ! LAST LINE IN SEARCH WINDOW
```

```
      ISWLP=IWINDP+ISWS-1      !  ,,  PIXEL ,,  ,,      ,,
      ITWFL=IWINDL+(ISWS-ITWS)/2  ! START LINE IN TEMPLATE WINDOW.
      ITWLL=ISWLL-(ISWS-ITWS)/2   !  LAST  ,,  ,,  ,,      ,,
      ITWFP=IWINDP+(ISWS-ITWS)/2  ! START PIXEL IN TEMPLATE WINDOW
      ITWLP=ISWLP-(ISWS-ITWS)/2   !  LAST  ,,  ,,  ,,      ,,
C
      SUMX=0.0          ! SUM TEMPLATE ELEMENTS FOR CALCULATION OF VARIANCE.
      SUMX2=0.0         ! SUM SQUARES OF TEMPLATE ELEMENTS.
C
C     3.0 START READING THE DATA.
C
      WRITE(5,110)IWINDL,IWINDP
110   FORMAT(' L=',I3,'    P=',I5)
      EVA1=EVA0+IWINDP*2
      NLCSW=0           ! COUNTER FOR NUMBER OF LAND/CLOUD PIXELS IN S.WINDOW
      NLCTW=0           ! SAME COUNTER FOR TEMPLATE WINDOW.
      KSR=0             ! ROW INDEX FOR SEARCH ARRAY - ISARR( ,KSR)
      KTR=0             !  ,,  ,,  ,,  TEMPLATE ,,  - ITARR( ,KTR)
C
C     3.1 READ LINES.
C
      DO 500 L=IWINDL,ISWLL
        KSR=KSR+1
        EVADDR=EVA1+L*E2WPL
        CALL ZMRVMA(EVADDR,ISWS,QQQ,IERR,IRVMA)
        IF(IERR.EQ.1)GOTO 140
        NAMMOD='ZMRVMA '
        GOTO 1000       ! RETURN WITH ERROR.
140     CONTINUE
C
C     3.2 EXTRACT PIXEL VALUE AND TURN INTO INTEGER.
C
C     WRITE(5,142)L,IWINDP,IFPPR,LIN1,ENWPL,ESTADR,EPDIF,E2WPL,EVA1,
C    +  EVADDR,IRVMA(1),BRVMA(1,1),BRVMA(2,1)
C142  FORMAT(' L=',I5,' IWINDP=',I5,/,
C    +  ' IFPPR,LIN1,ENWPL,ESTADR,EPDIF=',5I8,/,
C    +  ' E2WPL,EVA1,EVADDR=',3I8,/,
C    +  ' IRVMA(1),BRVMA(1,1),BRVMA(2,1)=',I8,2G6)
      IF(L.LT.ITWFL.OR.L.GT.ITWLL)THEN  ! OUTSIDE TEMPLATE WINDOW
         DO 200 I=1,ISWS
           BCON(1)=BRVMA(1,I)   ! SEARCH WINDOW DATA TO INTEGER
           IF(ICON.NE.255)GOTO 160  ! ICON FROM EQUIV. TEST FOR LAND/CLOUD
           NLCSW=NLCSW+1
           IF(NLCSW.GT.ISWLIM)GOTO 600 ! TOO MUCH LAND/CLOUD .
           ICON=0
160        ISARR(I,KSR)=ICON           ! STORE IN SEARCH ARRAY.
200      CONTINUE
        ELSE
         I=0              ! LINES INSIDE THE TEMPLATE REGION.
         ITC=0            ! TEMPLATE ARRAY COLUMN INDEX - ITARR(I,XTR)
         KTR=KTR+1        !    ,,     ,,    ROW     ,,        ,,
         DO 300 J=IWINDP,ISWLP
           I=I+1
           BCON(1)=BRVMA(1,I)           ! SEARCH WINDOW DATA TO INTEGER.
           IF(ICON.NE.255)GOTO 220      ! TEST FOR LAND/CLOUD IN S WINDOW
           NLCSW=NLCSW+1        ! COUNT NO. OF PIXELS WITH LAND/CLOUD.
           IF(NLCSW.GT.ISWLIM)GOTO 600 ! IF TOO MANY ABORT THIS WINDOW
           ICON=0
220        ISARR(I,KSR)=ICON           ! SAVE IN SEARCH ARRAY.
           IF(J.GE.ITWFP.AND.J.LE.ITWLP)THEN ! PIXELS INSIDE T-WINDOW
```

```
                ITC=ITC+1                    ! INCREMENT TEMPLATE ARRAY COLUMNS.
                BCON(1)=BRVMA(2,I)           ! CONVERT T-WINDOW DATA TO INTEGER
                IF(ICON.NE.255)GOTO 240 ! CHECK FOR LAND/CLOUD IN T-WINDOW
                NLCTW=NLCTW+1    ! COUNT PIXELS WITH LAND/CLOUD IN T-WINDOW
                IF(NLCTW.GT.ITWLIM)GOTO 600 ! IF TOO MANY ABORT THIS WINDOW
                ICON=0
240             ITARR(ITC,KTR)=ICON      ! STORE IN TEMPLATE ARRAY.
                SUMX=SUMX+FLOAT(ICON)
                SUMX2=SUMX2+(FLOAT(ICON))**2.0
              END IF
300           CONTINUE
            END IF
500       CONTINUE                    ! END OF THIS WINDOW
C
C         4.0 DO TEMPLATE MATCHING. CALCULATE START AND END LINE/PIXELS.
C             CALCULATE TEMPLATE VARIANCE. STORE IN XXXVEC.DAT
C
C         DO 520 I=1,ISWS
C           WRITE(3,510)(ISARR(J,I),J=1,ISWS)
C510        FORMAT(1X,20I4)
C520      CONTINUE
C         WRITE(3,530)
C530      FORMAT(1X//)
C         DO 540 I=1,ITWS
C           WRITE(3,510)(ITARR(J,I),J=1,ITWS)
C540      CONTINUE
          CALL CRCOEF(ITWS,ISWS,XXX,RMAXCC,IROW,ICOL)
C         WRITE(3,550)RMAXCC,IROW,ICOL
C550      FORMAT(1X/' RMAXCC,IROW,ICOL=',F8.3,2I7)
          IF(ABS(RMAXCC).GE.0.4)THEN
            IVSP=IWINDP+ISWS/2               ! VECTOR START PIXEL.
            IVEL=IWINDL+IROW-1               ! VECTOR END LINE.
            IVEP=IWINDP+ICOL-1               ! VECTOR END PIXEL.
            VAR=(RN2*SUMX2-(SUMX)**2.0)/RNP        ! TEMPLATE VARIANCE.
            WRITE(4,*)IVSL,IVSP,IVEL,IVEP,RMAXCC,VAR
            NUMV=NUMV+1                      ! COUNT THE VECTORS
          END IF
C
C         5.0 UPDATE WINDOW POSITION. GO DO NEXT WINDOW.
C
600       IWINDP=IWINDP+ISHIFT
          IF((IWINDP+ISWS-1).GT.ILPPR)THEN          ! END OF THIS LINE
            IWINDP=IPPPR             ! SET PIXEL COORDINATE FOR NEXT WINDOW
            IWINDL=IWINDL+ISHIFT     !  ,, LINE      ,,      ,,  ,,    ,,
            IVSL=IVSL+ISHIFT         ! SET VECTOR START LINE ( CENTRE OF NEW
                                     ! SEARCH WINDOW
          END IF
          IF((IWINDL+ISWS-1).LE.LIN2)GOTO 100
C
C
1000      RETURN
          END
```

```
C
C     SUBROUTINE CRCOEF.FTN
C     A ROUTINE CREATED FOR USE WITH PROGRAM AUTOTR - AUTOMATIC FEATURE
C     TRACKING FOR SEA SURFACE ADVECTION VECTORS.
C     CRCOEF DO TEMPLATE MATCHING BETWEEN TWO INTEGER ARRAYS, THE TEMPLATE
C     ITARR(32,32) AND THE ARRAY TO BE SEARCHED, ISARR(64,64).
C     THE BEST POSITION OF THE TEMPLATE WITHIN THE SEARCH ARRAY IS DETERMINED
C     BY THE HIGHEST CORRELATION COEFFICIENT RMAXCC.
C     THE COR. COEFFICIENT IS CALCULATED ACCORDING TO EQUATION NO.1 IN
C     TABLE 1 ,PAGE 143 OF SVEDLOV ET.AL. 1978.
C     THE ABSULUTE VALUE OF THE CORR. COEFF. IS <=1.0 . THIS SUBROUTINE
C     RETURNS THE LARGEST CORR. COEFF. ALONG WITH ITS ROW AND COLUMN
C     NUMBERS. NOTE !! : IF MORE THAN ONE POSITION YIELD THE SAME - LARGEST -
C     COEFF. THEN THE FIRST POSITION ENCOUNTERED WILL BE RETURNED.
C
C     !!! NB !!!  DIMENSIONS OF BOTH THE TEMPLATE AND THE SEARCH ARRAY
C     SHOULD BE ODD.
C
C
C     J.J.AGENBAG  S.F.R.I.  JUNE 1990.  FORTRAN 77
C
C
      SUBROUTINE CRCOEF(N,M,XXX,RMAXCC,IROW,ICOL)
      IMPLICIT COMPLEX(A),BYTE(B),CHARACTER*1(C),INTEGER*4(E)
      DIMENSION RCSTOR(36,2),RCWORK(36,2),RTSUM(3),RTWORK(2),RXY(2)
      COMMON/WINCOM/ISARR(64,64),ITARR(32,32)
C
C
      RN2=FLOAT(N*N)              ! N=DIMENSION OF TEMPLATE ARRAY.
      IS=1+N/2                    ! LEFT-MOST AND TOP-MOST POSITION OF THE CENTRE OF
                                  ! THE TEMPLATE IN THE SEARCH ARRAY IE WHERE DATA WILL
                                  ! BE WRITTEN IN THE ARRAY RCORC().
      IE=M-N/2                    ! M=DIMENSION OF SEARCH ARRAY. RIGHT-MOST AND BOTTOM-
                                  ! MOST POSITION OF THE CENTRE OF THE TEMPLATE.
      IP=N/2
C
C     1.0 DO SUMMATIONS OF TEMPLATE ARRAY. THESE SUMS WILL BE CONSTANT
C         FOR THIS TEMPLATE.
C
      SUMX=0.0                    ! SUM OF ALL TEMPLATE ELEMENTS.
      SUMX2=0.0                   ! SUM OF SQUARES OF TEMPLATE ELEMENTS.
      DO 100 I=1,N                ! THE ROWS
        DO 80 J=1,N               ! THE COLUMNS
          SUMX=SUMX+FLOAT(ITARR(J,I))
          SUMX2=SUMX2+(FLOAT(ITARR(J,I)))**2.0
80      CONTINUE
100   CONTINUE
      RC=RN2*SUMX2-(SUMX)**2.0          ! A CONSTANT IN THE COR. COEFF.
C
C
C     CALCULATION OF THE CORRELATION COEFFICIENTS FOR ALL POSITIONS OF
C     THE TEMPLATE WITHIN THE SEARCH ARRAY :
C     POSITION THE TEMPLATE IN THE TOP LEFT CORNER OF THE SEARCH ARRAY.
C     THEN CALCULATE THE FOLLOWING SUMS FOR THE NXN ARRAY UNDER THE TEMPLATE.
C       1. SUMY= SUM OF SEARCH ARRAY ELEMENTS. (X=TEMPLATE, Y=SEARCH ARRAY)
C       2. SUMY2=SUM OF Y**2
C       3. SUMXY=SUM OF X*Y
C     THE SUMS ARE CALCULATED FOR EACH COLUMN AND STORED IN RCSTOR(N,3)
C     AS WELL AS FOR THE TOTAL ARRAY RTSUM(3). RCSTOR IS COPIED TO
```

```
C       RCWORK(N,3) AND RTSUM() TO RTWORK().
C       THE COR. COEFF. IS CALCULATED FOR THIS POSITION. THE TEMPLATE IS THEN
C       SHIFTED TOWARDS THE RIGHT - ONE COLUMN AT A TIME. FOR EACH POSITION
C       THE NEW SUMS ARE OBTAINED BY DISCARDING THE SUMS FOR THE LEFT-MOST
C       COLUMN ( SUBTRACT RCWORK(1,) FROM RTWORK() ) AND BY ADDING A NEW
C       SET OF SUMS FOR THE NEW RIGHT-MOST COLUMN.  ! NOTE. THIS CAN NOT BE
C       DONE TO GET SUMXY. FOR SUMXY THE INTIRE WINDOW MUST BE SUMMED.
C       WHEN REACHING THE END OF THE ROW (THE TEMPLATE CENTRE AT IE) THEN
C       THE TEMPLATE IS SHIFTED IN ROW DOWN AND BACK TO THE LEFT (TEMPLATE
C       CENTRE AT IS). THE COLUMN SUMS IN RCSTOR() AND THE TOTAL SUMS,
C       RTSUM(), ARE UPDATED BY SUBTRACTION OF THE VALUES IN THE DISCARDED
C       TOP-MOST ROW AND BY ADDITION OF THE VALUES IN THE NEW BOTTOM ROW.
C       ! ONCE AGAIN  RTSUM(3) (=SUMXY) MUST BE CALCULATED FROM A FULL
C       SUMMATION OVER THE NEW WINDOW.
C
C
C       2.0 INITIALISE FOR TOP LEFT CORNER.
C
        DO 140 I=1,2                ! 1=SUMY,2=SUMY2 AND 3=SUMXY
          DO 120 J=1,N              ! THE COLUMNS
            RCSTOR(J,I)=0.0         ! ZERO THE COLUMN SUMS.
120       CONTINUE
          RTSUM(I)=0.0              ! ZERO THE OVERALL SUMS
140     CONTINUE
        RTSUM(3)=0.0
C
        RMAXCC=0.0        ! INITIALISE THE MAX. CORR. COEFF.
C
C
C       2.1  FOR THE TOP LEFT POSITION OBTAIN THE COLUMN SUMS. DO THE OVERALL
C            SUMS AT THE SAME TIME
C
        DO 200 J=1,N                ! THE ROWS
          DO 160 I=1,N              ! THE COLUMNS
            RXY(1)=FLOAT(ISARR(I,J))                ! =Y
            RXY(2)=RXY(1)**2.0                      ! =Y**2
            DO 156 K=1,2            ! 1=SUMY,2=SUMY2
              RCSTOR(I,K)=RCSTOR(I,K)+RXY(K)
              RTSUM(K)=RTSUM(K)+RXY(K)
156         CONTINUE
            RTSUM(3)=RTSUM(3)+RXY(1)*FLOAT(ITARR(I,J))
160       CONTINUE
200     CONTINUE
C
C       3.0 FOR THE LEFT-MOST POSITION IN THE ROW CALCULATE THE COR. COEFF.
C
        IR1=1     ! TOP ROW IN SEARCH ARRAY (FOR THIS WINDOW POSITION)
        IR2=N     ! BOTTOM ROW IN SEARCH ARRAY
        IR=IS     ! ROW POSITION OF CENTRE OF TEMPLATE ( IN ISARR() AND RCORC(
210     IC=IS     ! COLUMN POSITION OF CENTRE OF TEMPLATE.
        JC=N+1
        RSQTF=RC*(RN2*RTSUM(2)-RTSUM(1)**2.0)
        IF(RSQTF.LE.0.0)GOTO 215
        RCORC=(RN2*RTSUM(3)-SUMX*RTSUM(1))/SQRT(RSQTF)
        IF(ABS(RCORC).GT.ABS(RMAXCC))THEN
          RMAXCC=RCORC
          IROW=IR
          ICOL=IC
        END IF
C
```

```
C       4.0 TRANSFER THE CONTENTS OF RCSTOR() TO RCWORK() AND RTSUM() TO
C           TO RTWORK(). SUBTRACT COLUMN 1 SUMS FROM RTWORK()
C
215     DO 240 J=1,2
          RTWORK(J)=RTSUM(J)
          RTWORK(J)=RTWORK(J)-RCSTOR(1,J)
          DO 220 I=2,N
            RCWORK(I-1,J)=RCSTOR(I,J)
220       CONTINUE
240     CONTINUE
C
C       5.0 SHIFT THE WINDOW TOWARDS THE RIGHT - ONE COLUMN AT A TIME.
C
250     DO 360 IC=IS+1,IE
C         5.1 DO A NEW N'TH COLUMN SUMMATION
          RCWORK(N,1)=0.0
          RCWORK(N,2)=0.0
          K=0
          RTSUM(3)=0.0
          DO 280 I=IR1,IR2                ! THE ROWS IN THE COLUMN
            K=K+1
            RXY(1)=FLOAT(ISARR(JC,I))     ! Y
            RXY(2)=RXY(1)**2.0            ! Y**2
            DO 260 J=1,2
              RCWORK(N,J)=RCWORK(N,J)+RXY(J)
              RTWORK(J)=RTWORK(J)+RXY(J)
260         CONTINUE
C
C           5.1.1 SUM OVER THE ENTIRE WINDOW (ROWS IR1-IR2,COLUMS IC-IP TO
C                 IC+IP) FOR RTSUM(3) - IE SUMXY
C
            L=0
            DO 270 J=IC-IP,IC+IP
              L=L+1
              RTSUM(3)=RTSUM(3)+FLOAT(ISARR(J,I))*FLOAT(ITARR(L,K))
270         CONTINUE
280       CONTINUE
C
C         5.2 CALCULATE THE CORR. COEFF.
C
          RSQTF=RC*(RN2*RTWORK(2)-RTWORK(1)**2.0)
          IF(RSQTF.LE.0.0)GOTO 300
          RCORC=(RN2*RTSUM(3)-SUMX*RTWORK(1))/SQRT(RSQTF)
          IF(ABS(RCORC).GT.ABS(RMAXCC))THEN
            RMAXCC=RCORC
            IROW=IR
            ICOL=IC
          END IF
C
C         5.3 IF NOT AT THE END OF THE ROW THEN SUBTRACT COLUMN 1 SUMS FROM
C             RTWORK() AND SHIFT THE COLUMNS ONE OVER.
C
300       IF(IC.EQ.IE)GOTO 360            ! END OF THE ROW.
          DO 340 J=1,2
            RTWORK(J)=RTWORK(J)-RCWORK(1,J)
            DO 320 I=2,N
              RCWORK(I-1,J)=RCWORK(I,J)
320         CONTINUE
340       CONTINUE
          JC=JC+1
```

```
360     CONTINUE
C
C
C       6.0 AT THE END OF A ROW. GO DOWN ONE ROW AND BACK TO THE START POINT
C           SUBTRACT ROW IR1, Y AND Y**2 VALUES FROM THE RCSTOR() AND
C           RTSUM(). ADD THOSE FOR THE NEW LAST ROW -IR2+1
C
        IR=IR+1
        IF(IR.GT.IE)GOTO 500              ! NO MORE ROWS. DONE.
        IR2=IR2+1         ! INCREMENT WINDOW BOTTOM ROW INDEX.
        DO 400 I=1,N
          RXY(1)=FLOAT(ISARR(I,IR2))-FLOAT(ISARR(I,IR1))
          RXY(2)=(FLOAT(ISARR(I,IR2)))**2.0-(FLOAT(ISARR(I,IR1)))**2.0
          DO 380 K=1,2
            RCSTOR(I,K)=RCSTOR(I,K)+RXY(K)
            RTSUM(K)=RTSUM(K)+RXY(K)
380       CONTINUE
400     CONTINUE
        IR1=IR1+1         ! INCREMENT WINDOW TOP ROW INDEX.
C
C       6.1 DO A NEW SUMMATION FOR SUMXY OVER ROWS IR1-IR2 AND COLUMNS 1-N
C
        RTSUM(3)=0.0
        L=0
        DO 440 I=IR1,IR2
          L=L+1
          DO 420 J=1,N
            RTSUM(3)=RTSUM(3)+FLOAT(ISARR(J,I))*FLOAT(ITARR(J,L))
420       CONTINUE
440     CONTINUE
C
        GOTO 210
C
C
C
500     RETURN
        END
```

```
;
;       TASK BUILD INDIRECT COMMAND FILE FOR AUTOTR - AUTOMATIC FEATURE
;       TRACKING FOR DETERMINATION OF SEA SURFACE ADVECTION VECTORS.
;
AUTOTR/FP/CP=AUTOTR.ODL/MP
;
MAXBUF=512
COMMON=DPICOM:RW:5
UNITS=6
ASG=IM:1,DD1:2,SY.3,SY:4,TT0:5,TT1:6
;
;       GET REMAINING INFO FROM OVERLAY FILE, AUTOTR.ODL
;
```

```
;
;       OVERLAY DESCRIPTOR FILE FOR AUTOTR - AUTOMATIC FEATURE TRACKING
;       FOR DETERMINATION OF SEA SURFACE ADVECTION VECTORS.
;
        .ROOT AUTOTR-*(G1,G2,G3)
;
G1:     .FCTR AUTOT1-LB3
G2:     .FCTR AUTOT2-LB2
G3:     .FCTR AUTOT3-CRCOEF-LB2
;
LB1:    .FCTR DR:[100,300]NONRLS/LB-DR:[100,300]COMMON/LB-DR:[100,300]VMALIB/LB
LB2:    .FCTR DR:[100,300]VMALIB/LB
LB3:    .FCTR DR:[100,300]COMMON/LB
;
.END
;
```

APPENDIX E

Listing of program 'MTRACK'

```
C
C       PROGRAM MTRACK.FTN  - MANUAL FEATURE TRACKING.
C
C       THIS PROGRAM ASSUMES :
C       A) THAT THE IMAGE PROCESSING SYSTEM IS SET UP FOR THE SPLIT SCREEN
C          DISPLAY MODE.
C       B) THAT THE TWO IMAGES TO BE USED FOR FEATURE TRACKING, HAVE BEEN
C          TRANSFORMED TO MERCATOR PROJECTION (AND PREFERABLY CO-REGISTERED)
C       C) THAT THE TWO IMAGES HAVE BEEN LOADED NONDECIMATED AND THAT EACH
C          CONSIST OF A FEATURE AND ATLEAST ONE OF THEM SHOULD ALSO CONTAIN
C          ATLEAST ONE THEME.
C
C       THE MAIN PROGRAM CALLS 4 SUBROUTINES WHICH , IN TURN, CALL OTHER
C       SUBROUTINES AND/OR MODULES PRESENT IN THE ARIES II IMAGE PROCESSING
C       LIBRARIES.
C
C       A) MTRAC0 : PRESENTS THE MAIN MENU.
C
C       B) MTRAC1 : USED TO SELECT IMAGES,FEATURES AND THEMES AND TO MODIFY
C                   LOADED FEATURES. IT DEFINES MANY VARIABLES FOR USE
C                   BY THE REST OF THE PROGRAM.  4 SUBROUTINES :
C           1. READPT : READS THE IMAGE DISPLAY PARAMETER TABLE TO DEFINE
C                       VARIABLES RELATED TO THE LOADED IMAGES.
C           2. SELFEA : USER SELECT THE FEATURES TO BE USED FOR TRACKING
C           3. SELVD  : USER SELECT DISPLAY AND THEME FOR VECTOR DISPLAY AND
C                       DEFINE THE VECTOR SCALING AND POSITIONING. ALSO OPENS
C                       THE DIRECT ACCESS FILE FOR STORING VECTOR COORDINATES
C           4. MODFEA : MODIFY LOADED FEATURES IN THE IMAGES SELECTED FOR
C                       FEATURE TRACKING.
C
C       C) MTRAC2 : MAIN PROGRAM SECTION FOR VECTOR DEFINITION. 5 MAIN ROUTINES:
C           1. MT2MEN - INITIALISES THE DISPLAY AND PRESENT VECTOR DEFINITION
C                       MENUS
C           2. MT2DMS - TO DEFINE A VECTOR, TO MOVE CURSOR/WINDOWS AND TO CHANGE
C                       WINDOW SIZE
C           3. MT2ROT - TO ROTATE A WINDOW
C           4. MT2COM - COMPUTE VECTOR END POINTS USING A CORRELATION COEFF.
C           5. MT2AEX - 'ACCEPT' A VECTOR, ERASE A VECTOR AND EXIT
C
C         THESE ROUTINES CALL A FURTHER 8 SUBROUTINES FOR SPECIFIC TASKS :
C           1. MT2VEC - DEFINES VECTOR START END POINTS, MARK THESE POINTS
C                       AND DRAW WINDOWS. ALSO USED TO REMOVE THESE MARKS.
C           2. MT2ARR - COMPUTES AND DRAW ARROWS REPRESENTING VECTORS. ALSO
C                       ERASE ARROWS.
C           3. CWINDO - COMPUTE COORDINATES FOR A SQUARE WINDOW OR A CROSS
C           4. CARROW - COMPUTE COORDINATES FOR AN ARROW
C           5. DANDD  - DRAW OR DELETE A DOT, A WINDOW, A CROSS OR AN ARROW.
C           6. RWIND  - READ WINDOW DATA FROM DISK FEATURE FILES( NO ROTATION)
C           7. RTWIN  - RESAMPLE ROTATED WINDOW DATA FROM DISK FILES.
C           8. CRCOEF - COMPUTES THE MAXIMUM CORRELATION COEFFICIENT FOR TWO
C                       WINDOWS.
C
C       D) MTRAC3 : READ VECTOR COORDINATES STORED IN DISK FILE BY MTRAC2,
C                   COMPUTE ABSOLUTE SPEED AND DIRECTION AND LIST ON PRINTER.
C                   IT USES ONE SUBROUTINE :
C           1. LISTV  - DO THE ACTUAL COMPUTING AND LISTING.
C
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
```

```
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
        PROGRAM MTRACK
        IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
        DIMENSION IVC(2,2),IDISP(2,5),IWINS(2),ITSET(2),IXD(2),
     + ESTAD(2),ENWPL(2),IFEAT(5,2),IDAREA(2,2),ITAREA(2,2),RTRANS(6,2)
        PARAMETER (LUNDD=1,LUNID=2,LUNDF=3)
C
        DATA IDISP/1,0,2,0,1,0,0,0,0,0/, IWINS/13,23/
        DATA ITAREA/4000,4000,0,0/
        DATA IMF,IFLAG1,IFLAG2,SFAC,CPOS,IRSAV/0,0,0,1.0,'S',1/
C
C
C       1.0 GOTO MAIN MENU.
C
100     CALL MTRAC0(CIMF,IMF,IFLAG1,IRNUM,IRSAV)
        GOTO(200,200,200,300,400,500)IMF
        GOTO 100
C
C       2.0 GOTO MTRAC1
C
200     CALL MTRAC1(IFLAG1,CIMF,LUNDD,LUNID,LUNDF,QQQ,IVC,IDISP,IXD,
     + ESTAD,ENWPL,SFAC,CPOS,ITSET,IFEAT,IVDN,RTRANS,IVNUM,IRNUM,
     + IDAREA)
        GOTO 100
C
C       3.0 GOTO MTRAC2
C
300     CALL MTRAC2(CIMF,IFLAG2,IVC,IDISP,IFEAT,IWINS,ITSET,IXD,ESTAD,
     + ENWPL,LUNID,LUNDF,IVDN,SFAC,CPOS,IVNUM,IRNUM,ITAREA)
        GOTO 100
C
C       4.0 GOTO MTRAC3
C
400     CALL MTRAC3(CIMF,IRNUM,IRSAV,LUNDF,RTRANS,IDISP,IDAREA,ITAREA)
        GOTO 100
C
C       5.0 EXIT
C
500     CALL EXIT
        END
```

```
PIP>TI:=KEYBD.FTN
C
C       SUBROUTINE KEYBD.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       THE SUBROUTINE USES AN EXECUTIVE QIO CALL TO READ A KEYBOARD ENTRY
C       IT THEN DECODES THE ENTRY AND RETURN.
C
C       CALL KEYBD(CVAL,COUT,JVAL)
C       CVAL     CHAR*1   N=NUMERIC ENTRY, C=CONTROL CHARACTER, A=CAPITAL
C                         ALPHA, B=OTHER ALPHA
C       COUT       ,,     THE CHARACTER ENTERED
C       JVAL     INT*2    IF CVAL=N THEN JVAL = NUMBER 0-9
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
        SUBROUTINE KEYBD(CVAL,COUT,JVAL)
        IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
        DIMENSION JPAR(6),JOSB(2),BINP(2)
        EQUIVALENCE (BINP(1),JINP)
C
        CALL GETADR(JPAR(1),BINP(1))
        JPAR(2)=1
        CALL WTQIO(512,5,1,1,JOSB,JPAR,JDSW)            ! EXECUTIVE CALL
        JVAL=JINP                               ! JINP FROM EQUIVALENCE
        CVAL='N'                ! NUMERIC ENTRY
        IF(JOSB(2).EQ.0)THEN
          CVAL='C'              ! A CONTROL CHARACTER
          IF(JOSB(1).EQ.246)CVAL='Z' ! CTRL-Z
          IF(JOSB(1).EQ.3329)CVAL='E'! ENTER
         ELSE
          IF(JVAL.GE.65.AND.JVAL.LE.90)THEN
            CVAL='A'                        ! VALID CAPITAL ALPHA
           ELSE IF(JVAL.LT.48.OR.JVAL.GT.57)THEN
            CVAL='B'
          END IF
        END IF
        WRITE(COUT,20)BINP(1)           ! CONVERT BYTE TO CHARACTER
20      FORMAT(A1)
        JVAL=JVAL-48
C
        RETURN
        END
```

```
C
C       SUBROUTINE MTRAC0.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       MTRAC0 PRESENTS THE MAIN MENU FOR THE PROGRAM.
C
C       CALL MTRAC0(CIMF,IMF,IFLAG1,IRNUM,IRSAV)
C        CIMF    CHAR*1 MENU SELECTION :
C                       I = SPECIFY IMAGES AND FEATURES
C                       M = MODIFY FEATURES
C                       V = SPECIFY VECTOR DISPLAY
C                       D = DEFINE VECTORS
C                       S = LIST AND SAVE VECTORS
C                       X = EXIT
C
C        IMF     INT*2 NUMERICAL EQUIVALENT OF CIMF
C        IFLAG1   ,,   FLAG. 0=PARAMETERS NOT INITIALISED; 1=INITIALISED.
C        IRNUM    ,,   NUMBER OF RECORDS IN VECTOR STORAGE FILE - XXXVEC.DAT
C        IRSAV    ,,   NUMBER OF RECORDS WHEN DATA LAST SAVED BY MTRAC3.
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE MTRAC0(CIMF,IMF,IFLAG1,IRNUM,IRSAV)
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION CDATA(6)
      DATA CDATA/'I','M','V','D','S','X'/
      IF(IFLAG1.EQ.0)CIMF='I'
C
C
20    WRITE(5,40)CIMF
40    FORMAT(1X//1X,20X,'PROGRAM MTRACK :',/,1X,20X,16('-'),/,
     + 1X,10X,'SPECIFY IMAGES AND FEATURES [I]',/,
     + 1X,10X,'MODIFY FEATURES',13X,'[M]',/,
     + 1X,10X,'SPECIFY VECTOR DISPLAY',6X,'[V]',/,
     + 1X,10X,'DEFINE VECTORS',14X,'[D]',/,
     + 1X,10X,'LIST AND SAVE VECTORS',7X,'[S]',/,
     + 1X,10X,'EXIT',24X,'[X]',/,
     + 1X,10X,' SELECT [I/M/V/D/S/X] (',A1,'):',$)
      CALL KEYBD(CVAL,COUT,JVAL)      ! READ KEYBOARD INPUT
C
50    IF(CVAL.EQ.'Z')COUT='X'             ! CTRL-Z  = EXIT
      IF(CVAL.EQ.'E')COUT=CIMF               ! ACCEPT DEFAULT
      DO 60 J=1,6
        IF(COUT.EQ.CDATA(J))THEN
          CIMF=COUT
          IMF=J
          GOTO 100                          ! RETURN TO MAIN
        END IF
60    CONTINUE
      GOTO 20                               ! TRY AGAIN
C
100   IF(CIMF.EQ.'X')THEN
        IF(IRNUM.LE.IRSAV)GOTO 300          ! NOTHING NEW TO SAVE.
110     WRITE(5,120)
120     FORMAT(1X/' SAVE VECTORS BEFORE EXITTING ? [Y/N] (Y):',$)
        CALL KEYBD(CVAL,COUT,JVAL)      ! READ KEYBOARD INPUT.
```

```
140       IF(CVAL.EQ.'Z')GOTO 20                ! CTRL-Z
          IF(CVAL.EQ.'E')COUT='Y'               ! ACCEPT DEFAULT
          IF(COUT.EQ.'N')GOTO 300               ! EXIT
          IF(COUT.EQ.'Y')THEN
            CIMF='S'
            IMF=5
            GOTO 300                            ! GO SAVE. EXIT
          END IF
          GOTO 110                              ! TRY AGAIN
        END IF
C
C
300     RETURN
        END
```

```
C
C       SUBROUTINE MTRAC1.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       MTRAC1 IS CALLED TO SPECIFY THE IMAGES, FEATURES AND THEMES TO BE
C       USED IN THE FEATURE TRACKING OPERATION. IT MAY ALSO BE CALLED TO
C       CHANGE THE FEATURES ALREADY LOADED - EG. APPLY A CONTRAST STRETCH.
C
C       THIS ROUTINE DEFINES MOST OF THE PARAMETERS REQUIRED BY OTHER PARTS OF
C       MTRACK AND SHOULD BE CALLED AS PART OF THE INITIALISATION.
C
C       CALL MTRAC1(IFLAG1,CIMF,LUNDD,LUNID,LUNDF,QQQ,IVC,IDSP,IXD,ESTAD,ENWPL,
C                   SFAC,CPOS,ITSET,IFEAT,IVDN,RTRANS,IVNUM,IRNUM,IDAREA)
C
C       IFLAG1          INT*2 PARAMETER INITIALISATION FLAG 0= NOT INITIALISED
C       CIMF            CHAR*1 MAIN MENU FUNCTION.
C       LUNDD           INT*2  LOGICAL UNIT NUMBER ASSIGNED TO THE VIDEO SUBSYSTEM
C       LUNID             ,,     ,,     ,,    ,,      ,,    ,,  ,, IMAGE DISK.
C       LUNDF             ,,     ,,     ,,    ,,      ,,    ,,  ,, DATA FILE.
C       IVC(2,2)          ,,    CURSOR COORDINATES : (1,)=LINE  (2,)=PIXEL
C                               (1,)=TEMPLATE (2,)=SEARCH IMAGE DISPLAY
C       IDISP(2,5)        ,,    (1,1) & (2,1) = TEMPLATE IMAGE AND FEATURE NO.
C                               (1,2) & (2,2) = SEARCH   IMAGE AND FEATURE NO.
C                               (1,3) & (2,3) = VECTOR IMAGE AND THEME NUMBER
C                               (1,4) & (2,4) = TEMPLATE DISPLAY START LINE AND PIXE
C                               (1,5) & (2,5) = SEARCH IMAGE DISPLAY START LINE/PIXE
C       IXD(2)            ,,    IXEL DEPTH  (1)=TEMPLATE, (2)=SEARCH IMAGE DISPLAY
C       ESTAD(2)        INT*4   START ADDRES FOR IMAGE IN VIDEO MEMORY 1=TEMPLATE
C                               2=SEARCH IMAGE DISPLAY
C       ENWPL(2)          ,,    NUMBER OF WORDS PER DISPLAY LINE. 1=TEMPLATE
C                               2=SEARCH IMAGE DISPLAY.
C       SFAC            REAL*4  SCALING FACTOR FOR VECTOR REPRESENTATION
C       CPOS            CHAR*1  RELATIVE POSITION FOR VECTOR ARROWS
C       ITSET(2)        INT*2   VECTOR THEME BIT MASK AND SET/DELETE VALUE.
C       IFEAT(5,2)        ,,    (,1)=TEMPLATE, (,2)=SEARCH IMAGE FEATURE FILE
C                               (1,) & (2,)= HIGH AND LOW ORDER BYTES FOR DISK STOR-
C                               AGE ADDRES; (3,)= NUMBER OF PIXELS PER SCAN LINE;
C                               (4,) & (5,)= FIRST LINE AND PIXEL IN THE FILE.
C       IVDN              ,,    DISPLAY USED FOR DRAWING VECTORS 1=ON TEMPLATE
C                               2= ON SEARCH IMAGE DISPLAY
C       RTRANS(6,2)     REAL*4  IMAGE GEOMETRIC TRANSFORM DATA (,1)= TEMPLATE IMAGE,
C                               (,2)= SEARCH IMAGE DISPLAY
C                               (1,)&(2,)= A REFERENCE POINT LAT. AND LONG.
C                               (3,)&(4,)= LINE/PIXEL NUMBERS FOR THE REFERENCE PT.
C                               (5,)= SCALING FACTOR (NO. OF PIXELS=1 MIN OF LONG)
C                               (6,1)= TIME LAG BETWEEN IMAGES (HOURS)
C                               (6,2)= PRINTER INITIALISATION FLAG (0= NOT DONE)
C       IVNUM           INT*2   NUMBER OF VECTORS IN STORAGE FILE
C       IRNUM             ,,    NUMBER OF RECORDS IN VECTOR STORAGE FILE.
C       IDAREA(2,2)       ,,    DISPLAY AREA. (1,)&(2,)=LINE AND PIXEL NUMBERS.
C                               (,1)=TOP LEFT CORNER; (,2)=BOTTOM RIGHT CORNER.
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE MTRAC1(IFLAG1,CIMF,LUNDD,LUNID,LUNDF,QQQ,IVC,IDISP,
     + IXD,ESTAD,ENWPL,SFAC,CPOS,ITSET,IFEAT,IVDN,RTRANS,IVNUM,IRNUM,
     + IDAREA)
```

```
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IVC(2,2),IDISP(2,5),IXD(2),ESTAD(2),ENWPL(2),ITSET(2),
     + IFEAT(5,2),RTRANS(6,2),IDAREA(2,2),
     + BAR(3,2),BFNAM(9,2),BFEA(4,3,2),BTH(4,8,2),NFEA(2),NLP(2,2),
     + NTHM(2),CTHPB(2),NBFEA(3,2),JSUMT(2),JTHM(4,16,2),BIMD(4),
     + JFCS(2,3,2)
C
      CHARACTER CMOD*7,CSUB*7
      DATA BIMD/'I','M','A','G'/
C
C     1.0 INITIALISE
C
      IERR=1
      CSUB='MTRAC1 '
      IF(IFLAG1.EQ.0)THEN                    ! MUST INITIALISE
        CALL ZIGVID(LUNDD,QQQ,IERR)          ! ATTACH VIDEO SUBSYSTEM
        IF(IERR.NE.1)THEN
          CMOD='ZIGVID '
          CIMF='X'        ! FATAL ERROR. ABORT
          GOTO 4000
        END IF
C
        CALL ZKDLNK(LUNID,BIMD,'D',QQQ,IERR)  ! LINK TO THE IMAGE DISK.
        IF(IERR.NE.1)THEN
          CMOD='ZKDLNK '
          CIMF='X'
          GOTO 4000       ! FATAL ERROR. EXIT.
        END IF
        GOTO 300
       ELSE
        IF(CIMF.EQ.'I')GOTO 300
        IF(CIMF.EQ.'V')JGO=550
        IF(CIMF.EQ.'M')JGO=800
        GOTO 520
      END IF
C
C     2.0 SELECT IMAGES
C
300   J=1
320   IF(J.EQ.1)WRITE(5,340)IDISP(1,J)
      IF(J.EQ.2)WRITE(5,350)IDISP(1,J)
340   FORMAT(1X/' ENTER TEMPLATE IMAGE NUMBER(',I2,' CTRL-Z):',$)
350   FORMAT(1X/' ENTER SEARCH IMAGE NUMBER(',I2,' CTRL-Z):',$)
      CALL KEYED(CVAL,COUT,JVAL)        ! COMMON ROUTINE TO DECODE KEY INPUT
360   IF(CVAL.EQ.'E')GOTO 380  ! ACCEPT DEFAULT
      IF(CVAL.EQ.'N'.AND.JVAL.GT.0.AND.JVAL.LT.7)GOTO 370 ! VALID INPUT
      IF(CVAL.EQ.'Z')THEN
         IF(J.EQ.2)GOTO 300
         IF(IFLAG1.EQ.1)THEN
           CIMF='D'
          ELSE
           CIMF='X'
         END IF
         GOTO 5000       ! RETURN TO MAIN MENU
      END IF
      GOTO 320           ! INVALID INPUT
370   IDISP(1,J)=JVAL
380   J=J+1
```

```
      IF(J.LE.2)GOTO 320
C
C     3.0 ENTER TRANFORMATION REFERENCE DATA. USED BY LISTV TO COMPUTE
C         ABSOLUTE SPEEDS AND DIRECTIONS FOR VECTORS.
C
400   DO 460 J=1,2
        IF(J.EQ.2)THEN
         WRITE(5,404)
404      FORMAT(1X/' SEARCH IMAGE DATA THE SAME AS TEMPLATE IMAGE ?',
     +   ' [Y/N](Y):',$)
         CALL KEYBD(CVAL,COUT,JVAL)
         IF(CVAL.EQ.'Z')GOTO 300                ! CTRL-Z.   STEP BACK
         IF(CVAL.EQ.'E'.OR.(CVAL.EQ.'A'.AND.COUT.EQ.'Y'))THEN
           DO 406 K=1,5
            RTRANS(K,2)=RTRANS(K,1)
406        CONTINUE
           GOTO 460
         END IF
        END IF
C
        IF(J.EQ.1)WRITE(5,410)
410     FORMAT(1X//' IN ORDER TO CONVERT VECTORS TO ABSOLUTE VALUES',
     +  ' THE',/,
     +  ' LATITUDE AND LONGITUDE FOR A POINT IN THE IMAGE IS REQUIRED'/
     +  ' ALSO THE SCALING FACTOR ( NO. OF PIXELS=1 MIN OF LONG) AND'/
     +  ' THE TIME LAG BETWEEN THE TWO IMAGES.')
        IF(J.EQ.1)WRITE(5,420)
        IF(J.EQ.2)WRITE(5,425)
420     FORMAT(1X//' FOR THE TEMPLATE IMAGE ENTER:')
425     FORMAT(1X//' FOR THE SEARCH IMAGE ENTER :')
        WRITE(5,430)
430     FORMAT(1X/' ENTER REFERENCE LINE AND PIXEL :',$)
        READ(5,*)RTRANS(3,J),RTRANS(4,J)
        WRITE(5,440)
440     FORMAT(' ENTER THE REF. LAT AND LONG(-=S AND E):',$)
        READ(5,*)RTRANS(1,J),RTRANS(2,J)
        WRITE(5,450)
450     FORMAT(' ENTER SCALING FACTOR ( NO. OF PIXELS=1 MIN OF LONG):',
     +  $)
        READ(5,*)RTRANS(5,J)
460   CONTINUE
      WRITE(5,470)
470   FORMAT(1X/' ENTER THE TIME LAG BETEEN THE TWO IMAGES(HOURS):',$)
      READ(5,*)RTRANS(6,1)
C
C     4.0 CALL SUBROUTINE READPT TO READ AND DECODE THE IMAGE PARAMETER
C         TABLES FOR THE SELECTED IMAGES.
C
500   JGO=550
520   CALL READPT(IDISP,QQQ,IXD,ESTAD,ENWPL,BAR,BTH,BFEA,NFEA,NLP,
     + NBFEA,JFCS,NTHM,JTHM,JSUMT,CTHPB,IERR,CMOD,CSUB,JGO,JIMG)
      J=JIMG
      IF(JGO.EQ.320)GOTO 320     ! STEP BACK
      IF(JGO.EQ.800)GOTO 800     ! GO MODIFY FEATURES.
      IF(JGO.EQ.4000)GOTO 4000   ! EXIT WITH ERROR
      IF(IFLAG1.NE.0.AND.CIMF.EQ.'V')GOTO 700
C
C     5.0 CALL SUBROUTINE SELFEA TO SELECT FEATURES.
C
600   JGO=650
```

```
      CALL SELFEA(IDISP,LUNID,BAR,NFEA,BFEA,NBFEA,QQQ,
     + IVC,IFEAT,BFNAM,JGO)
      IF(JGO.EQ.300)GOTO 300
650   IF(IFLAG1.EQ.1.AND.CIMF.EQ.'I')GOTO 5000
C
C     6.0 CALL SUBROUTINE SELVD TO SELECT THE IMAGE AND THEME FIELD
C         FOR VECTOR DISPLAY.
C
700   JGO=750
      ITF1=IFLAG1                          ! LOCAL FLAG
      IF(ITF1.EQ.0)IVDN=0
      ITF2=IVDN                            ! LOCAL FLAG
      CALL SELVD(IFLAG1,CIMF,LUNDF,IDISP,NTHM,BAR,BTH,JTHM,
     + JSUMT,CTHPB,QQQ,IVDN,CPOS,SFAC,ITSET,IVNUM,IRNUM,JGO)
      IF(JGO.EQ.600)GOTO 600       ! STEP BACK
      IF(ITF1.EQ.0.OR.ITF2.NE.IVDN)THEN        ! DEFINE THE DISPLAY AREA.
        IDAREA(1,1)=IDISP(1,3+IVDN)
        IDAREA(2,1)=IDISP(2,3+IVDN)
        IDAREA(1,2)=IDAREA(1,1)+NLP(1,IVDN)-1
        IDAREA(2,2)=IDAREA(2,1)+NLP(2,IVDN)-1
      END IF
750   GOTO 5000
C
C     7.0 CALL SUBROUTINE MODFEA TO MODIFY A LOADED FEATURE.
C
800   CALL MODFEA(IDISP,BAR,NFEA,JFCS,NBFEA,BFEA,QQQ,IERR,CMOD,CSUB)
      IF(IERR.EQ.1)GOTO 5000               ! RETURN
C
C     8.0 COMMON EXIT WITH ERROR
C
4000  WRITE(5,4020)IERR,CMOD,CSUB
4020  FORMAT(1X//' ERROR NO.',I5,' REPORTED BY ',A7,'IN SUBROUTINE ',
     + A7,//)
      GOTO 5000
C
C
5000  RETURN
      END
```

```
C
C       SUBROUTINE READPT.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING
C
C       THIS ROUTINE USES THE ARIES II MODULE ZMIPAR TO READ THE PARAMETER
C       TABLES FOR THE IMAGES SELECTED AS TEMPLATE AND SEARCH IMAGES. THE
C       BINARY DATA ARE DECODED AND RETURNED TO MTRACI TO SELECT FEATURES,
C       THEMES, ETC.
C
C       J.J.AGENEAU  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
      SUBROUTINE READPT(IDISP,QQQ,IXD,ESTAD,ENWPL,BAR,BTH,EFEA,
     + NFEA,NLP,NBFEA,JFCS,NTHM,JTHM,JSUMT,CTHPB,IERR,CMOD,CSUB,
     + JCO,JIMC)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IDISP(2,5),IXD(2),ESTAD(2),ENWPL(2),BAR(3,2),
     + BTH(4,8,2),BFEA(4,3,2),NFEA(2),NLP(2,2),NBFEA(3,2),JFCS(2,3,2),
     + NTHM(2),JTHM(4,16,2),JSUMT(2),CTHPB(2),NPTH(8),JTIP(8),
     + BY2(2),BY4(4)
      CHARACTER CMOD*7,CSUB*7
      COMMON /ZMIIPR/BIPAR(0:275)
      EQUIVALENCE (EY2(1),JEQ),(EY4(1),EEQ)
      IERR=1
      CSUB='READPT '
C
C       1.0 READ AND DECODE THE IMAGE PARAMETER TABLE FOR BOTH SPECIFIED
C           IMAGES IDISP(1,I) AND IDISP(1,2)
C
      DO 200 J=1,2
        CALL ZMIPAR(IDISP(1,J),QQQ,IERR)    ! ARIES II MODULE
        IF(IERR.NE.1)THEN
          IF(IERR.EQ.-300)THEN
            WRITE(5,20)IDISP(1,J)
20          FORMAT(1X//' IMAGE NO.',I2,' NOT LOADED. RESPECIFY PSE.')
            JIMC=J
            JCO=320
            GOTO 500
           ELSE
            CMOD='ZMIPAR '
            CIMF='I'
            JCO=4000
            GOTO 500          ! EXIT
          END IF
        END IF
C
C       1.1 DECODE RELEVANT PARTS OF THE TABLE
C
        JEQ=0
        EEQ=0
        EY2(1)=BIPAR(1)
        IXD(J)=JEQ                          ! IXEL DEPTH FROM EQUIVALENCE
C
        DO 40 K=1,3
          BAR(K,J)=BIPAR(K+1)               ! IMAGE AREA CODE
40      CONTINUE
C
```

```
      BY2(1)=BIPAR(10)
      BY2(2)=BIPAR(11)
      IDISP(1,J+3)=JEQ                  ! IMAGE START LINE
C
      BY2(1)=BIPAR(12)
      BY2(2)=BIPAR(13)
      IDISP(2,J+3)=JEQ                  ! IMAGE START PIXEL
C     BY2(1)=BIPAR(14)
      BY2(2)=BIPAR(15)
      NLP(1,J)=JEQ                      ! NUMBER OF LINES
C
      BY2(1)=BIPAR(16)
      BY2(2)=BIPAR(17)                  ! GET NO. OF PIXELS PER LINE
      NLP(2,J)=JEQ                      ! SAVE NUMBER OF PIXELS
      ENWPL(J)=JEQ                      ! NO. OF WORDS PER LINE. 15-BIT IXEL
      IF(IXD(J).EQ.7)ENWPL(J)=ENWPL(J)/2    ! 7-BIT IXEL
C
      DO 60 K=1,4
        BY4(K)=BIPAR(K+17)
60    CONTINUE
      ESTAD(J)=EEQ                      ! START ADDRES IN VIDEO MEMORY
C
      JEQ=0
      BY2(1)=BIPAR(22)
      NFEA(J)=JEQ                       ! NUMBER OF FEATURES
      IF(NFEA(J).NE.0)IDISP(2,J)=1   ! SET DEFAULT FEATURE NUMBER.
C
      BY2(1)=BIPAR(23)
      NTHM(J)=JEQ                       ! NUMBER OF THEMES
      CTHPB(J)='P'                      ! SET FOR 'PACKED THEMES'
      BY2(1)=BIPAR(134)
      IF(JEQ.EQ.66)CTHPB(J)='B'     ! SET TO 'BIT SPECIFIC'
C
      K=23                              ! GET FEATURE FILE ID-CODES
      IF(NFEA(J).EQ.0)GOTO 100
      DO 80 L=1,NFEA(J)
        K=24+(L-1)*8-1
        DO 70 M=1,4
          BFEA(M,L,J)=BIPAR(K+M)    ! ID FOR FEATURE L
70      CONTINUE
        BY2(1)=BIPAR(K+6)
        NBFEA(L,J)=JEQ                  ! NO. OF BITS ALLOCATED TO THIS FEATURE.
C
        BY2(1)=BIPAR(K+7)
        JFCS(2,L,J)=JEQ                 ! DISPLAY SATURATION VALUE.
        BY2(1)=BIPAR(K+8)
        JFCS(1,L,J)=JEQ                 ! DISPLAY CUTOFF VALUE.
80    CONTINUE
C
C     4.1.1 THE THEMES
C
      IF(NTHM(J).EQ.0)GOTO 200      ! NO THEMES LOADED
      K=K+8
100   L=1
      JSUMT(J)=0
110   JSUMC=0
      DO 130 M=1,4
        K=K+1
        BTH(M,L,J)=BIPAR(K)         ! FILE FROM WHICH THEME WAS LOADED
        IF(BIPAR(K).EQ."0)JSUMC=JSUMC+1
```

```
130       CONTINUE
          IF(JSUMC.GE.2)THEN
            L=L-1
            GOTO 200
          END IF
C
          JEQ=0
          K=K+2
          BY2(1)=BIPAR(K)
          JMSK=JEQ                    ! THEME BIT MASK
C
          K=K+1
          BY2(1)=BIPAR(K)
          NFTH(L)=JEQ                 ! NO. OF THEMES FROM THIS FILE
          K=K+1
          JSUMT(J)=JSUMT(J)+NFTH(L)
C
          N=0
          M=8
          JMV=JMSK
140       IF((2**(M-1)).GT.JMV)GOTO 150
          N=N+1
          JTIF(N)=M
          JMV=JMV-2**(M-1)
          IF(JMV.EQ.0)GOTO 160
150       M=M-1
          GOTO 140
160       N=0
          DO 180 M=(JSUMT(J)-NFTH(L)+1),JSUMT(J)
            N=N+1
            JTHM(1,M,J)=L
            JTHM(2,M,J)=JTIF(NFTH(L)-N+1)
            JTHM(3,M,J)=JMSK
            JTHM(4,M,J)=M*2
180       CONTINUE
C
          IF(JSUMT(J).GE.NTHM(J))GOTO 200
          L=L+1
          GOTO 110
200     CONTINUE
C
C
500     RETURN
        END
```

```
C
C       SUBROUTINE SELFEA.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       THIS SUBROUTINE IS CALLED BY MTRAC1 TO SELECT THE FEATURES (FEATURE
C       FILES) TO BE USED IN THE FEATURE TRACKING PROCEDURE. THIS IS
C       NECESSARY WHEN WINDOWS AND CORRELATION COEFFICIENTS ARE USED IN THE
C       TRACKING.
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
        SUBROUTINE SELFEA(IDISP,LUNID,BAR,NFEA,BFEA,NBFEA,QQQ,IVC,
     + IFEAT,BFNAM,JGO)
C
        IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
        DIMENSION IDISP(2,5),BAR(3,2),NFEA(2),BFEA(4,3,2),NBFEA(3,2),
     + IVC(2,2),IFEAT(5,2),BFNAM(9,2),BFDUM(9),BFILN(9)
C
C       1.0 SELECT FEATURES
C
20      DO 200 J=1,2                    ! 1=TEMPLATE, 2=SEARCH IMAGE
          IF(J.EQ.1)WRITE(5,30)
          IF(J.EQ.2)WRITE(5,40)
30        FORMAT(1X//' SELECT FEATURE FOR THE TEMPLATE IMAGE :',$)
40        FORMAT(1X//' SELECT FEATURE FOR THE SEARCH IMAGE :',$)
          IF(NFEA(J).EQ.0)THEN
            WRITE(5,50)
50          FORMAT(1X/' NO FEATURES LOADED FOR THIS IMAGE.')
            GOTO 200
          END IF
          WRITE(5,60)IDISP(1,J),(BAR(K,J),K=1,3),
     +    ((K,(BFEA(L,K,J),L=1,4),NBFEA(K,J)),K=1,NFEA(J))
60        FORMAT(1X/' IMAGE NO.',I2,'  AREA CODE : ',3A1,/,
     +    ' FEATURE NO.    FILE ID.    NUMBER OF BITS',/,
     +    (1X,I6,10X,4A1,5X,I7,/))
80        WRITE(5,90)IDISP(2,J)
90        FORMAT(1X/' ENTER FEATURE NO.(',I2,' CTRL-Z):',$)
          CALL KEYBD(CVAL,COUT,JVAL)        ! COMMON ROUTINE TO DECODE KEY INPUT
100       IF(CVAL.EQ.'E')GOTO 110    ! ACCEPT DEFAULT
          IF(CVAL.EQ.'N')THEN
            IF(JVAL.GT.0.AND.JVAL.LE.NFEA(J))THEN ! VALID INPUT
              IDISP(2,J)=JVAL
            ELSE                                  ! INVALID INPUT
              GOTO 80
            END IF
           ELSE IF(CVAL.EQ.'Z')THEN               ! CTRL-Z
            IF(J.EQ.2)GOTO 20
            JGO=300
            GOTO 300                              ! BACK TO SELECT IMAGES
           ELSE
            GOTO 80                               ! TRY AGAIN
          END IF
C
C         1.1  CONSTRUCT THE 9-CHARACTER FILE NAME.
C
110       BFNAM(4,J)='F'
```

```
          BFNAM(5,J)='F'        ! STANDARD CODE FOR A FEATURE FILE.
          DO 120 K=1,3
            BFNAM(K,J)=BAR(K,J)
120       CONTINUE
          DO 130 K=1,4
            BFNAM(K+5,J)=BFEA(K,IDISP(2,J),J)
130       CONTINUE
          DO 135 K=1,9
            BFILN(K)=BFNAM(K,J)
135       CONTINUE
C
C         1.2 OBTAIN THE DISK ADDRES FOR THIS FILE
C
          CALL ZFSEEK(BFILN,1,QQQ,IERR,K,IFEAT(1,J),IFEAT(2,J),
     +    BFDUM)
          IF(IERR.NE.1)THEN
           CMOD='ZFSEEK '
           WRITE(5,140)IERR,CMOD,BFILN
140        FORMAT(1X//' ERROR NO.',I5,' REPORTED BY ',A7,'FOR ',9A1,/,
     +     ' FEATURE DATA UNDEFINED.')
           IDISP(2,J)=0
           GOTO 200
          END IF
C
C         1.3 OBTAIN IMAGE FILE DIMENSIONS.
C
          CALL ZHRDIM(BFILN,LUNID,QQQ,IERR,K,IFEAT(3,J),IFEAT(5,J),
     +    IFEAT(4,J),N,M)       ! ARIES II MODULE
          IF(IERR.NE.1)THEN
            CMOD='ZHRDIM '
            WRITE(5,140)IERR,CMOD,BFILN
            IDISP(2,J)=0
          END IF
C
C         1.4 INITIALISE CURSOR COORDINATES TO IMAGE CENTRE
C
          IVC(1,J)=IFEAT(4,J)+K/2
          IVC(2,J)=IFEAT(5,J)+IFEAT(3,J)/2
200     CONTINUE
C
C       WRITE(6,240)IDISP,LUNID,BAR,NFEA,BFEA,NBFEA,IVC,IFEAT,BFNAM
C240    FORMAT(1X//' IDISP=',6I3,4I5,'  LUNID=',I2,/,
C     + ' BAR=',3A1,2X,3A1,'   NFEA=',2I4,/,' BFEA=',6(4A1,2X),/,
C     + ' NBFEA=',6I3,'   IVC=',4I6,/,
C     + ' IFEAT=',10I6,/,' BFNAM=',9A1,3X,9A1,/)
C
300     RETURN
        END
```

```
C
C       SUBROUTINE SELV0.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       THIS SUBROUTINE IS CALLED BY MTRAC1 TO SELECT THE DISPLAY (TEMPLATE
C       OR SEARCH IMAGE) AND THE THEME FIELD TO BE USED FOR DISPLAYING THE
C       ADVECTION VECTORS (AS ARROWS). IT ALSO OBTAINS A SPECIFICATION AS TO
C       THE SCALE (SIZE) AND RELATIVE POSITION OF THE ARROWS. VECTORS ARE
C       DEFINED BY INDICATING THE START POSITION OF A FEATURE ( IN THE
C       TEMPLATE OR FIRST IMAGE) AND THE END POSITION (IN THE SEARCH OR 2ND
C       IMAGE). THE VECTOR IS REPRESENTED BY AN ARROW OF WHICH THE SIZE IS A
C       FRACTION OR MULTIPLE OF THE DISTANCE BETWEEN  THE TWO POSITIONS, AND
C       IT MAY BE DRAWN SO AS TO START (FEATHER SIDE OF ARROW) ON THE VECTOR
C       START POINT OR THE ARROW CENTRE MAY BE ON THE VECTOR START POINT.
C
C       THE SUBROUTINE ALSO OPENS A DIRECT ACCESS FILE FOR STORING THE VECTOR
C       AND ARROW COORDINATES (IN LINE/PIXEL COORDINATES), LATER TO BE USED FOR
C       COMPUTATION OF THE ASSOLUTE POSITION (LAT./LONG.) AND SIZE (SPEED AND
C       DIRECTION) OF THE VECTORS.
C
C
C       J.J.AGENDAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE SELVD(IFLAG1,CIMF,LUNDF,IDISP,NTHM,BAR,BTH,JTHM,
     + JSUMT,CTHPB,QQQ,IVDN,CPOS,SFAC,ITSET,IVNUM,IRNUM,JGO)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IDISP(2,5),NTHM(2),BAR(3,2),BTH(4,8,2),JTHM(4,16,2),
     + JSUMT(2),CTHPB(2),ITSET(2),JTMD(8)
      CHARACTER CDFILN*10
      DATA JTMD/2,6,6,14,14,14,14,30/
C
C
C       1.0 SELECT IMAGE AND THEME FOR VECTOR DISPLAY.
C
C       1.1 SELECT IMAGE
C
20      IF(NTHM(1).EQ.0.AND.NTHM(2).EQ.0)THEN
          WRITE(5,40)
40        FORMAT(1X//' NO THEMES LOADED IN EITHER IMAGE - CANNOT',
     +    ' DISPLAY VECTORS.')
          CIMF='X'
          JGO=5000              ! RETURN TO MENU
          GOTO 400
         ELSE
          IDISP(1,3)=IDISP(1,1)    ! SET DEFAULT FOR TEMPLATE IMAGE
          CHAR='T'
          IF(IDISP(1,3).EQ.0)THEN ! SET DEFAULT FOR SEARCH IMAGE
            IDISP(1,3)=IDISP(1,2)
            CHAR='S'
          END IF
        END IF
C
60      WRITE(5,80)CHAR
80      FORMAT(1X//' IMAGE FOR VECTOR DISPLAY :',/,
     + ' TEMPLATE OR SEARCH IMAGE ? CT/S/CTRL-Z)(',A1,'):',$)
```

```
      CALL KEYBD(CVAL,COUT,JVAL)          ! COMMON ROUTINE TO DECODE KEY INPUT
100   IF(CVAL.EQ.'E')GOTO 120   ! ACCEPT DEFAULT
      IF(CVAL.EQ.'A')THEN        ! VALID ALPHA INPUT
        IF(COUT.NE.'T'.AND.COUT.NE.'S')GOTO 60    ! INVALID
        IDISP(1,3)=IDISP(1,1)
        IF(COUT.EQ.'S')IDISP(1,3)=IDISP(1,2)
       ELSE IF(CVAL.EQ.'Z')THEN
        IF(IFLAC1.EQ.0)JGO=600 ! STEP BACK
        GOTO 400                 ! RETURN
       ELSE
        GOTO 60                  ! INVALID INPUT. TRY AGAIN
      END IF
C
C     1.2 CHECK THAT THERE ARE INFACT THEMES LOADED FOR SELECTED IMAGE.
C
120   J=1
      IF(IDISP(1,3).EQ.IDISP(1,2))J=2
      IVDN=J                     ! DISPLAY TO USE FOR VECTORS
      IF(NTHM(J).EQ.0)THEN
        WRITE(5,140)
140     FORMAT(1X/' NO THEMES LOADED FOR THIS IMAGE.')
        GOTO 60
      END IF
C
C     1.3 SELECT THEME FOR VECTOR DISPLAY.
C
160   WRITE(5,180)IDISP(1,3),(BAR(K,J),K=1,3),
     + ((L,(BTH(K,JTHM(1,L,J),J),K=1,4),JTHM(2,L,J)),L=1,JSUMT(J))
180   FORMAT(1X//' IMAGE NO.',I2,'  AREA CODE : ',3A1,/,
     + ' LOADED THEME NO    THEME FILE THEME NUMBER IN FILE',/,
     + (1X,I8,13X,4A1,12X,I4,/))
      IF(IDISP(2,3).EQ.0)IDISP(2,3)=1 ! SET DEFAULT
190   WRITE(5,200)IDISP(2,3)
200   FORMAT(' SELECT THEME NUMBER : (',I2,' CTRL-Z ):',$)
      CALL KEYBD(CVAL,COUT,JVAL)          ! COMMON ROUTINE TO DECODE INPUT.
220   IF(CVAL.EQ.'E')GOTO 240          ! ACCEPT DEFAULT
      IF(CVAL.EQ.'N')THEN              ! NUMERIC INPUT
        IF(JVAL.GT.0.AND.JVAL.LE.JSUMT(J))THEN
          IDISP(2,3)=JVAL              ! VALID SELECTION
         ELSE
          GOTO 190                     ! INVALID
        END IF
       ELSE IF(CVAL.EQ.'Z')THEN        ! CTRL-Z
        GOTO 20                        ! STEP BACK. SELECT IMAGE
       ELSE
        GOTO 190                       ! TRY AGAIN
      END IF
C
C     1.4 COMPUTE A MASK AND VALUE TO SET OR DELETE THE THEME (VECTOR SYMBOL
C
240   ITSET(1)=JTMD(JSUMT(J))          ! PACKED THEMES
      IF(CTHPB(J).EQ.'B')THEN          ! BIT SPECIFIC THEMES
        ITSET(1)=0
        DO 250 K=1,JSUMT(J)
          ITSET(1)=ITSET(1)+2**K
250     CONTINUE
      END IF
      ITSET(2)=IDISP(2,3)*2          ! PACKED THEMES
      IF(CTHPB(J).EQ.'B')ITSET(2)=2**IDISP(2,3)   ! BIT SPECIFIC THEMES
C
```

```
C       1.5 DETERMINE HOW VECTOR ARROWS ARE TO BE DRAWN.
C
        WRITE(5,260)
260     FORMAT(1X//' VECTOR REPRESENTATION AS ARROWS :',/,
     + ' 1. SCALING FACTOR DETERMINE ARROW LENGTH AS A FRACTION OR ',
     + 'MULTIPLE OF THE',/,'    DISTANCE BETWEEN INDICATED START AND ',
     + 'END POINTS.',//,
     + ' 2. CPOS DETERMINE POSITION RELATIVE TO INDICATED START POINT:'/
     + '     S=ARROW ORIGIN PLACED ON INDICATED START POINT.',/,
     + '     C=ARROW IS CENTRED ON THE INDICATED START POINT.')
        WRITE(5,280)
280     FORMAT(1X/' ENTER THE SCALING FACTOR :',$)
        READ(5,*)SFAC
        CPOS='S'
290     WRITE(5,300)CPOS
300     FORMAT(1X/' ENTER RELATIVE POSITION [S/C](',A1,'):',$)
        CALL KEYBD(CVAL,COUT,JVAL)   ! COMMON ROUTINE TO DECODE INPUT
320     IF(CVAL.EQ.'E')GOTO 340       ! ACCEPT DEFAULT
        IF(CVAL.EQ.'A'.AND.(COUT.EQ.'S'.OR.COUT.EQ.'C'))THEN
          IF(COUT.EQ.'C')CPOS='C'
          GOTO 340
         ELSE
          GOTO 290      ! INVALID INPUT
        END IF
C
C       2.0 SET IFLAG1 TO INDICATE DAT HAVE BEEN INITIALISED AND OPEN A DIRECT
C           ACCESS FILE FOR STORAGE OF VECTOR DATA
C
340     CIMF='D'        ! SET MENU FUCTION FOR VECTOR DEFINITION
        CDFILN(4:10)='VEC.DAT'   ! FILE NAME OF FORM : XXXVEC.DAT WHERE
        DO 360 J=1,3             ! XXX = AREA CODE OF VECTOR IMAGE
          WRITE(CHAR,350)BAR(J,IVDN)
350       FORMAT(A1)
          CDFILN(J:J)=CHAR
360     CONTINUE
        IF(IFLAG1.EQ.1)CLOSE(LUNDF)    ! IF FILE ALREADY OPEN, FIRST CLOSE
        IVNUM=0                        ! INITIALISE NUMBER OF VECTORS
        IRNUM=0                        ! INIT. NUMBER OF DATA RECORDS IN FILE
        CALL ERRSET(29,.TRUE.,.FALSE.,.TRUE.,.FALSE.,15)
        OPEN(UNIT=LUNDF,FILE=CDFILN,STATUS='OLD',ACCESS='DIRECT',
     + RECORDTYPE='FIXED',RECL=64,FORM='FORMATTED',ERR=390)
370     WRITE(5,375)CDFILN
375     FORMAT(1X/' CONTINUE USING EXISTING FILE ',A10,' FOR VECTOR',
     + ' STORAGE ? [Y/N](Y):',$)
        CALL KEYBD(CVAL,COUT,JVAL)
        IF(CVAL.EQ.'E'.OR.(CVAL.EQ.'A'.AND.COUT.EQ.'Y'))THEN
          J=1                          ! CHECK NUMBER OF RECORDS/VECTORS IN FILE
          CALL ERRSET(39,.TRUE.,.FALSE.,.TRUE.,.FALSE.,15)
380       READ(LUNDF,REC=J,FMT=385,END=389,ERR=388)N,N1,N2,N3,N4,N5,N6,
     +    N7,N8
385       FORMAT(1X,9I6)
          IRNUM=IRNUM+1
          IF(N.NE.0)IVNUM=IVNUM+1      ! DELETED VECTORS HAS A 0 IN THIS FIELD
          J=J+1
          GOTO 380
        END IF
        CLOSE(LUNDF)
        GOTO 390
388     IVNUM=IVNUM-1
        IRNUM=IRNUM-1
```

```
      GOTO 392
C
390   OPEN(UNIT=LUNDF,FILE=CDFILN,STATUS='NEW',ACCESS='DIRECT',
     + RECORDTYPE='FIXED',RECL=64,FORM='FORMATTED')
392   IFLAG1=1
C
C
400   RETURN
      END
```

```
C
C       SUBROUTINE MODFEA.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       THIS ROUTINE USES THE ARIES II MODULE ZIFEAT TO MODIFY THE LOADED
C       FEATURES IN THE IMAGES SELECTED FOR FEATURE TRACKING.
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991.  IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM.
C
C
      SUBROUTINE MODFEA(IDISP,BAR,NFEA,JFCS,NBFEA,BFEA,QQQ,
     + IERR,CMOD,CSUB)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IDISP(2,5),BAR(3,2),NFEA(2),JFCS(2,3,2),NBFEA(3,2),
     + BFEA(4,3,2),BFILN(9)
      CHARACTER CMOD*7,CSUB*7
C
      IERR=1
      CSUB='MODFEA '
C
C       1.0 FEATURE TO MODIFY.
C
      J=1
      CMOD(1,1)='T'
10    WRITE(5,20)CMOD(1,1)
20    FORMAT(1X//' MODIFY TEMPLATE OR SEARCH FEATURE ? (T/S/CTRL-Z)',
     + '(',A1,'):',$)
      CALL KEYED(CVAL,COUT,JVAL)          ! COMMON ROUTINE TO DECODE INPUT
40    IF(CVAL.EQ.'E')GOTO 50     ! ACCEPT DEFAULT
      IF(CVAL.EQ.'Z')GOTO 1000    ! EXIT
      IF(CVAL.EQ.'A')THEN         ! ALPHA INPUT
        IF(COUT.NE.'T'.AND.COUT.NE.'S')GOTO 10    ! TRY AGAIN
        J=1
        IF(COUT.EQ.'S')J=2
       ELSE
        GOTO 10                   ! INVALID INPUT
      END IF
C
C       2.0 FEATURE TO MODIFY.
C
50    WRITE(5,60)IDISP(1,J),(BAR(K,J),K=1,3),
     + ((K,(BFEA(L,K,J),L=1,4),NBFEA(K,J),JFCS(1,K,J),
     + JFCS(2,K,J)),K=1,NFEA(J))
60    FORMAT(1X/' IMAGE NO.',I2,'   AREA CODE : ',3A1,/,
     + ' FEATURE NO.    FILE ID.    NUMBER OF BITS    ',
     + 'CUTOFF    SATURATION',/,
     + (1X,I6,10X,4A1,5X,I7,10X,I4,5X,I7,/))
      JF=IDISP(2,J)
80    WRITE(5,90)JF
90    FORMAT(1X/' ENTER FEATURE NO.(',I2,' CTRL-Z):',$)
      CALL KEYBD(CVAL,COUT,JVAL)          ! COMMON ROUTINE TO DECODE KEY INPUT
100   IF(CVAL.EQ.'E')GOTO 110  .          ! ACCEPT DEFAULT
      IF(CVAL.EQ.'N')THEN
        IF(JVAL.GT.0.AND.JVAL.LE.NFEA(J))THEN ! VALID INPUT
           JF=JVAL
         ELSE                                 ! INVALID INPUT
```

```
          GOTO 80
        END IF
       ELSE IF(CVAL.EQ.'Z')THEN                    ! CTRL-Z
        GOTO 10                                    ! BACK TO SELECT IMAGE
       ELSE
        GOTO 80                                    ! TRY AGAIN
      END IF
C
C     3.0  CONSTRUCT THE 9-CHARACTER FILE NAME.
C
110   BFILN(4)='F'
      BFILN(5)='F'        ! STANDARD CODE FOR A FEATURE FILE.
      DO 120 K=1,3
        BFILN(K)=BAR(K,J)
120   CONTINUE
      DO 130 K=1,4
        BFILN(K+5)=BFEA(K,JF,J)
130   CONTINUE
C
C     4.0 ENTER CUT-OFF AND SATURATION VALUES.
C
160   WRITE(5,180)JFCS(1,JF,J)
180   FORMAT(' ENTER CUT OFF VALUE [',I3,'] :',$)
      READ(5,*)M
      IF(M.LT.0.OR.M.GT.255)THEN
        WRITE(5,190)
190     FORMAT(1X/' VALUE OUT OF RANGE.')
        GOTO 160
      END IF
200   WRITE(5,210)JFCS(2,JF,J)
210   FORMAT(' ENTER SATURATION VALUE [',I3,'] :',$)
      READ(5,*)N
      IF(N.LT.M.OR.N.GT.255)THEN
        WRITE(5,190)
        GOTO 200
      END IF
C
C     5.0 DO THE MODIFICATION.
C
      JFCS(1,JF,J)=M
      JFCS(2,JF,J)=N
      CALL ZIFEAT(IDISP(1,J),IDISP(2,J),'O','F',QQQ,IERR,SSS,
     + N,M,BFILN)
      IF(IERR.NE.1)THEN
        CMOD='ZIFEAT '
        GOTO 1000
      END IF
      GOTO 10
C
C     9.0  RETURN
C
1000  RETURN
      END
```

```
C
C     SUBROUTINE MTRAC2.FTN
C      PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C     THIS IS THE MAIN PROGRAM SECTION FOR DEFINITION OF VECTORS.
C
C     CALL MTRAC2(CIMF,IFLAG2,IVC,IDISP,IWINS,ITSET,IXD,ESTAD,ENWPL,
C                 SFAC,CPOS,LUNID,LUNDF,IFEAT,IVDN,QQQ,ITAREA,IVNUM,
C                 IRNUM)
C
C     CIMF           CHAR*1 MAIN MENU FUNCTION.
C     IFLAG2         INT*2  DISPLAY INITIALISATION FLAG. 0=NOT INITIALISED.
C     IVC(2,2)         ,,   CURSOR COORDINATES : (1,)=LINE  (2,)=PIXEL
C                           (,1)=TEMPLATE (,2)=SEARCH IMAGE DISPLAY
C     IDISP(2,5)       ,,   (1,1) & (2,1) = TEMPLATE IMAGE AND FEATURE NO.
C                           (1,2) & (2,2) = SEARCH   IMAGE AND FEATURE NO.
C                           (1,3) & (2,3) = VECTOR IMAGE AND THEME NUMBER
C                           (1,4) & (2,4) = TEMPLATE DISPLAY START LINE AND PIXEL
C                           (1,5) & (2,5) = SEARCH IMAGE DISPLAY START LINE/PIXEL
C     IWINS(2)         ,,   WINDOW SIZES (1)=TEMPLATE, (2)=SEARCH AREA
C     ITSET(2)         ,,   VECTOR THEME BIT MASK AND SET VALUE
C     IXD(2)           ,,   DISPLAY IXEL DEPTH (1)=TEMPLATE AND (2) SEARCH IMAGE
C     ESTAD(2)       INT*4  START ADDRES FOR IMAGE IN VIDEO MEMORY 1=TEMPLATE
C                           2=SEARCH IMAGE DISPLAY
C     ENWPL(2)         ,,   NUMBER OF WORDS PER DISPLAY LINE. 1=TEMPLATE
C                           2=SEARCH IMAGE DISPLAY.
C     SFAC           REAL*4 SCALING FACTOR TO DETERMINE SIZE OF ARROW REPRE-
C                           SENTING THE VECTOR EG. 2=TWICE INDICATED LENGTH ETC.
C     CPOS           CHAR*1 WHERE TO DRAW ARROW. 'S'=ARROW START ON INDICATED
C                           VECTOR START POINT; 'C'=ARROW CENTRED ON START POINT.
C     LUNID          INT*2  LOGICAL UNIT NUMBER ASSIGNED TO THE IMAGE DISK
C     LUNDF            ,,     ,,     ,,     ,,     ,,     ,,  ,, DATA FILE.
C     IFEAT(5,2)       ,,   (,1)=TEMPLATE, (,2)=SEARCH IMAGE FEATURE FILE
C                           (1,) & (2,)= HIGH AND LOW ORDER BYTES FOR DISK STOR-
C                           AGE ADDRES; (3,)= NUMBER OF PIXELS PER SCAN LINE;
C                           (4,) & (5,)= FIRST LINE AND PIXEL IN THE FILE.
C     IVDN             ,,   DISPLAY USED FOR DRAWING THE VECTORS 1=TEMPLATE
C                           2=SEARCH IMAGE DISPLAY
C     ITAREA(2,2)      ,,   PART OF IMAGE TO SAVE. (1,) & (2,)= LINE AND PIXEL
C                           (,1)=TOP LEFT CORNER; (,2)=BOTTOM RIGHT CORNER.
C     IVNUM            ,,   NUMBER OF VECTORS EXTRACTED
C     IRNUM            ,,   NUMBER OF RECORDS WRITTEN TO THE VECTOR STORAGE
C                           FILE.
C
C
C      J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C      IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
C
      SUBROUTINE MTRAC2(CIMF,IFLAG2,IVC,IDISP,IFEAT,IWINS,ITSET,
     + IXD,ESTAD,ENWPL,LUNID,LUNDF,IVDN,SFAC,CPOS,IVNUM,IRNUM,ITAREA)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IVC(2,2),IDISP(2,5),IFEAT(5,2),IWINS(2),ITSET(2),
     + IXD(2),ESTAD(2),ENWPL(2),ITAREA(2,2),
     + JMAXWS(2),JMINWS(2),JDOTF(2),JWINF(2),IVCOOR(2,4),IWCOOR(2,5,2)
C
      DATA JMAXWS,JMINWS,JDOTF,JWINF,JVECF,IRA/17,27,5,18,0,0,0,0,0,0/
```

```
      CDV1='C'
C
C     1.0 CALL MT2MEN TO INITIALISE DISPLAY AND/OR PRESENT MENU.
C
50    CDV2='D'
100   CALL MT2MEN(CIMF,IFLAG2,CDV1,CDV2,IVC,IDISP)
      IF(CIMF.EQ.'X')GOTO 1000          ! RETURN.
C
C     2.0 CALL MT2DMS TO DEFINE VECTORS, TO MOVE CURSOR/WINDOW OR
C         TO CHANGE WINDOW SIZE.
C
      IF(CDV2.EQ.'D'.OR.CDV2.EQ.'M'.OR.CDV2.EQ.'S')THEN
        CALL MT2DMS(CDV1,CDV2,IVC,IDISP,IWINS,
     +  IXD,ESTAD,ENWPL,LUNDF,IVDN,SFAC,CPOS,IVNUM,IRNUM,ITAREA,
     +  JVECF,JDOTF,JWINF,JMAXWS,JMINWS,IRA,IVCOOR,IWCOOR)
        GOTO 100
C
C       3.0 CALL MT2ROT TO ROTATE A WINDOW
C
       ELSE IF(CDV2.EQ.'R')THEN
        CALL MT2ROT(CDV1,CDV2,IVC,IDISP,IWINS,
     +  IXD,ESTAD,ENWPL,ITAREA,JWINF,IRA,IWCOOR)
        GOTO 100
C
C       4.0 CALL MT2COM TO COMPUTE THE END POINT FOR A VECTOR USING
C           WINDOWS AND MAXIMUM CORRELATION COEFFICIENT.
C
       ELSE IF(CDV2.EQ.'C')THEN
        CALL MT2COM(CDV1,CDV2,IVC,IDISP,IFEAT,IWINS,
     +  IXD,ESTAD,ENWPL,LUNID,LUNDF,IVDN,SFAC,CPOS,IVNUM,IRNUM,ITAREA,
     +  JVECF,JDOTF,JWINF,IRA,IVCOOR,IWCOOR)
        GOTO 100
C
C       CALL MT2AEX TO ACCEPT OR ERASE A VECTOR OR TO EXIT.
C
       ELSE
        CALL MT2AEX(CDV1,CDV2,IVC,IDISP,IWINS,ITSET,
     +  IXD,ESTAD,ENWPL,LUNDF,IVDN,SFAC,CPOS,IVNUM,IRNUM,ITAREA,
     +  JVECF,JDOTF,JWINF,IRA,IVCOOR,IWCOOR)
        GOTO 100
C
C       6.0 EXIT
C
      END IF
1000  RETURN
      END
```

```
C
C       SUBROUTINE MT2MEN.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       THIS SUBROUTINE IS CALLED BY MTRACK TO INITIALISE THE DISPLAY AND/OR
C       TO PRESENT THE VECTOR DEFINE SUB MENUS.
C
C
C       J.J.AGENDAG  S.F.P.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
C
      SUBROUTINE MT2MEN(CINF,IFLAG2,CDV1,CDV2,IVC,IDISP)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*1(I),INTEGER*2(I-P),
     + REAL*8(G-H)
      DIMENSION IVC(2,3),IDISP(2,3),CDATA(8)
      CHARACTER*7 CMCD,CSUB
      DATA CDATA/'D','M','S','R','C','A','E','X'/
C
      IERR=1
      CSUB='MT2MEN '
C
C     1.0 IF IFLAG2=0 THEN INITIALISE DISPLAY.
C
      IF(IFLAG2.EQ.0)THEN
       DO 100 J=2,1,-1
         CALL ZMDISP(J,IDISP(1,J),-1,GGG,IERR)      ! LINK IMAGE TO DISPLAY
         IF(IERR.NE.1)THEN
          CMCD='ZMDISP '
          GOTO 300                                  ! REPORT ERROR. RETURN
         END IF
C
         CALL ZMMCUR(IVC(1,J),IVC(2,J),1,GGG,IERR) ! POSITION CURSOR
         IF(IERR.NE.1)THEN
           CMCD='ZMMCUR '
           GOTO 300                                 ! REPORT ERROR. RETURN
         END IF
C
         CALL ZMDISP(-1,-1,2,GGG,IERR)
100     CONTINUE
      END IF
C
C     2.0 SELECT MODE OF OPERATION IE. CURSOR ON WINDOW
C
      IF(CDV2.NE.'X'.AND.IFLAG2.NE.0)GOTO 300
200   WRITE(5,220)CDV1
220   FORMAT(1X//' CURSOR OR WINDOW MOVE ? CR/W/CTRL-Z? (',A1,') ',$)
      CALL KEY3D(CVAL,COUT,JVAL)      ! READ/DECODE KEY INPUT
      IF(CVAL.EQ.'E')GOTO 280         ! ACCEPT DEFAULT
      IF(CVAL.EQ.'Z')THEN             ! CTRL-Z.  EXIT
         CINF='X'
         GOTO 400
       ELSE IF(CVAL.EQ.'A')THEN       ! ALPHA INPUT
        IF(COUT.EQ.'C'.OR.COUT.EQ.'W')THEN  ! CORRECT INPUT
          CDV1=COUT
          GOTO 280
        END IF
        GOTO 200                      ! INVALID INPUT
```

```
          ELSE
            GOTO 200                        !     ,,      ,,
          END IF
260       IFLAG2=1                          ! SET FLAG TO SHOW INITIALISATION DONE.
          CDV2='D'
C
C         3.0 SELECT FUNCTION CDV2
C
300       WRITE(5,320)CDV2
320       FORMAT(1X//1X,10X,'DEFINE VECTOR',7X,'[D]',/,
         + 1X,10X,'MOVE CURSOR/WINDOW  [M]',/,
         + 1X,10X,'CHANGE WINDOW SIZE  [S]',/,
         + 1X,10X,'ROTATE WINDOW',7X,'[R]',/,
         + 1X,10X,'COMPUTE VECTOR',6X,'[C]',/,
         + 1X,10X,'ACCEPT VECTOR',7X,'[A]',/,
         + 1X,10X,'ERASE VECTOR',8X,'[E]',/,
         + 1X,10X,'EXIT',16X,'[X]',//,
         + ' SELECT FUNCTION [D/M/S/R/C/A/E/X/CTRL-Z] (',A1,'):',$)
          CALL KEYBD(CVAL,COUT,JVAL)          ! READ/ DECODE KEY INPUT
          IF(CVAL.EQ.'E')COUT=CDV2            ! ACCEPT DEFAULT
          IF(CVAL.EQ.'Z')COUT='X'             ! CTRL-Z.  TAKE AS EXIT
          IF(CVAL.EQ.'B')GOTO 300             ! INVALID INPUT. TRY AGAIN
          DO 340 J=1,8
            IF(COUT.EQ.CDATA(J))THEN
              CDV2=COUT
              GOTO 600                        ! RETURN
            END IF
340       CONTINUE
          GOTO 300                            ! TRY AGAIN
C
C         4.0 EXIT WITH ERROR
C
500       WRITE(5,520)IERR,CMOD,CSUB
520       FORMAT(1X//' ERROR NO.',I5,' REPORTED BY ',A7,'SUBROUTINE ',
         + A7,//)
          CIMP='X'
C
C         5.0 RETURN
C
600       RETURN
          END
```

```
C
C       SUBROUTINE MT2DMS.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       MT2DMS IS CALLED BY MTRAC2 TO DEFINE VECTORS, TO MOVE THE CURSOR/
C       WINDOW OR TO CHANGE WINDOW SIZES.
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
        SUBROUTINE MT2DMS(CDV1,CDV2,IVC,IDISP,IWINS,IXD,ESTAD,ENWPL,
     + LUNDF,IVDN,SFAC,CPOS,IVNUM,IRNUM,ITAREA,JVECF,JDOTF,JWINF,
     + JMAXWS,JMINWS,IRA,IVCOOR,IWCOOR)
C
        IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
        DIMENSION IVC(2,2),IDISP(2,5),IWINS(2),IXD(2),ESTAD(2),ENWPL(2),
     + ITAREA(2,2),JDOTF(2),JWINF(2),JMAXWS(2),JMINWS(2),IVCOOR(2,4),
     + IWCOOR(2,5,2),ITSET(2)
C
        IERR=1
        IF(CDV2.NE.'D')GOTO 200
C
C       1.0 DEFINE VECTORS. CDV2='D'
C
20      JS=1
        JE=2
C
C       1.1 ERASE EXISTING VECTOR (ANNOTATION FIELD) - IF ANY.
C
40      IF(JVECF.EQ.1)THEN                ! JVECF= FECTOR FLAG
          CFLD='A'                        ! A=ANNOTATION FIELD
          CSD='D'                         ! SET/DELETE FUNCTION = DELETE
          CALL MT2AR1(IVC,IDISP,ITSET,IXD,ESTAD,ENWPL,LUNDF,IVDN,
     +    SFAC,CPOS,IVNUM,IRNUM,ITAREA,JVECF,IVCOOR,CFLD,CSD)
        END IF
C
C       1.2 USE MT2VEC TO DEFINE START AND END POINTS
C
60      JQES=1                          ! PROMPT FLAG = YES
        IF(CDV2.EQ.'S')JQES=0
        JDRF=1                          ! DRAW FLAG = YES
        CALL MT2VC1(CDV1,IVC,IDISP,IWINS,IXD,ESTAD,ENWPL,ITAREA,JDOTF,
     + JWINF,IRA,IWCOOR,JS,JE,JQES,JDRF,QQQ,IERR)
        IF(JERR.NE.1)GOTO 500           ! RETURN WITH ERROR
C
C       1.3 IF IN WINDOW MODE SET DEFAULT FUNCTION FOR COMPUTE VECTOR(CDV2='C')
C           IF IN CURSOR MODE, DRAW THE VECTOR IN ANNOTATION FIELD AND SET
C           CDV2='A' (ACCEPT VECTOR)
C
70      IF(CDV1.EQ.'W')THEN             ! WINDOW MODE
          CDV2='C'
          GOTO 500                      ! RETURN
        ELSE                            ! CURSOR MODE. REMOVE MARKS AND DRAW
          JQES=0                        ! VECTOR IN ANNOTATION FIELD
          JDRF=0
          CALL MT2VC1(CDV1,IVC,IDISP,IWINS,IXD,ESTAD,ENWPL,ITAREA,JDOTF,
     +    JWINF,IRA,IWCOOR,JS,JE,JQES,JDRF,QQQ,IERR)
          IF(IERR.NE.1)GOTO 500
```

```
30        IF(JE.NE.2)GOTO 500
          CFLD='A'                    ! SET PARAMETERS TO DRAW THE
          CSD='S'                     ! ARROW
          JDIRF=1
          CDV2='A'
          CALL NT2AR1(IVC,IDISP,ITSET,IXD,ESTAD,ENWPL,LUNDF,IVDN,SFAC,
     +    CPOS,IVNUM,IRNUM,ITAREA,JVECF,IVCOOR,CFLD,CSD)
          GOTO 500                    ! RETURN
        END IF
C
C       2.0 MOVE CURSOR/WINDOW OR CHANGE WINDOW SIZE
C
200     CHAR='T'
210     WRITE(5,220)CHAR
220     FORMAT(1X//' TEMPLATE OR SEARCH WINDOW ? (T/S/CTRL-Z) (',A1,
     + ') :',$)
        CALL KEYBD(CVAL,COUT,JVAL)  ! READ/DECODE KEY INPUT
        IF(CVAL.EQ.'Z')GOTO 500     ! RETURN
        IF(CVAL.EQ.'E')COUT=CHAR    ! ACCEPT DEFAULT
        IF(CVAL.EQ.'B')GOTO 210     ! TRY AGAIN
        IF(COUT.EQ.'T')THEN
          JS=1
          GOTO 230
         ELSE IF(COUT.EQ.'S')THEN
          JS=2
          GOTO 230
         ELSE
          GOTO 210                    ! TRY AGAIN
        END IF
C
230     JE=JS
        IF(CDV2.EQ.'M')GOTO 40      ! GO MOVE THE WINDOW/CURSOR
C
C       2.1 CHANGE WINDOW SIZE
C
        IF(CDV1.EQ.'C')THEN
          WRITE(5,300)
300       FORMAT(1X//' FUNCTION NOT APPLICABLE TO CURSOR MODE.')
          CDV2='D'
          GOTO 500
         ELSE
320       WRITE(5,340)IWINS(JS),JMAXWS(JS)
340       FORMAT(1X/' PRESENT WINDOW SIZE=',I3,/,
     +    ' ENTER NEW SIZE (MAX=',I3,' MUST BE ODD):',$)
          READ(5,*)JINP
          IF(IMOD(JINP,2).EQ.0)GOTO 320          ! MUST BE ODD
          IF(JINP.GE.JMINWS(JS).AND.JINP.LE.JMAXWS(JS))THEN
            IWINS(JS)=JINP
            JMINWS(2)=JMINWS(1)+1
            GOTO 40                                  ! GO CHANGE WINDOW SIZE ON
           ELSE                                      ! THE SCREEN
            GOTO 320                                 ! INVALID INPUT
          END IF
        END IF
C
C       3.0 RETURN
C
500     RETURN
        END
```

```
C
C       SUBROUTINE MT2ROT.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       MT2ROT IS CALLED BY MTRAC2 TO ROTATE WINDOWS.
C
C       J.J.AGENDAC  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
        SUBROUTINE MT2ROT(CDV1,CDV2,IVC,IDISP,IWINS,IXD,ESTAD,ENWPL,
     + ITAREA,JWINF,IRA,IWCOOR)
C
        IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
        DIMENSION IVC(2,2),IDISP(2,5),IWINS(2),IXD(2),ESTAD(2),
     + ENWPL(2),ITAREA(2,2),JWINF(2),IWCOOR(2,5,2),
     + JCOOR(2,5),JCF(2)
C
        JCF(1)=0                        ! 'CROSS FLAG' 0= NO CROSS ON DISPLAY
        JCF(2)=0                        !  1 OR 2
C
C       1.0 DELETE WINDOW AND REPLACE BY A CROSS HAIR TO DETERMINE ROTATION
C           ANGLE (ROTATE ON SEARCH IMAGE DISPLAY). THEN REMOVE CROSS HAIR
C           AND REPLACE WINDOWS.
C
20      IF(CDV1.EQ.'C')THEN             ! FUNCTION NOT FOR CURSOR MODE
          WRITE(5,40)
40        FORMAT(1X//' FUNCTION NOT APPLICABLE TO CURSOR MODE.',/)
          GOTO 500                      ! RETURN
        END IF
C
        IF(JWINF(1).EQ.0.AND.JWINF(2).EQ.0)THEN     ! NO WINDOWS DEFINED
          WRITE(5,60)
60        FORMAT(1X//' NO WINDOWS DEFINED. CANNOT EXECUTE.',/)
          GOTO 500
        END IF
C
C       1.1 SET PARAMETERS TO DELETE BOTH WINDOWS.
C
        CSDW='D'
        JS=1
        JE=2
80      DO 100 J=JS,JE
          JROTA=0
          IF(J.EQ.2)JROTA=IRA
          IF(CSDW.EQ.'S')THEN           ! BEFORE RESETTING WINDOWS MUST COMPUTE
            CALL CWIND4(IVC(1,J),IVC(2,J),IWINS(J),FLOAT(JROTA),
     +       'W',QQQ,JCOOR)
            DO 84 K=1,5
              IWCOOR(1,K,J)=JCOOR(1,K)
              IWCOOR(2,K,J)=JCOOR(2,K)
84          CONTINUE
           ELSE
            DO 90 K=1,5
              JCOOR(1,K)=IWCOOR(1,K,J)
              JCOOR(2,K)=IWCOOR(2,K,J)
90          CONTINUE
          END IF
```

```
      CALL DANDD(IDISP(1,J+3),IDISP(2,J+3),IXD(J),ESTAD(J),ENWPL(J),
     +  ITAREA,1,1,'A',CSDW,'W',JCOOR)
      JWINF(J)=1-JWINF(J)                    ! IF 0 SET TO 1 AND VV
100   CONTINUE
C
      IF(CSDW.EQ.'S')GOTO 500                ! RETURN
C
C     1.1 SET PARAMETERS TO DRAW TWO CROSSES.
C
      CSDC='S'
110   DO 120 J=JS,JE
        IF(CSDC.EQ.'S'.AND.JCF(J).NE.0)THEN     ! DELETE EXISTING CROSS
         CALL DANDD(IDISP(1,J+3),IDISP(2,J+3),IXD(J),ESTAD(J),
     +   ENWPL(J),ITAREA,1,1,'A','D','C',JCOOR)
         JCF(J)=0
        END IF
        JROTA=0
        IF(J.EQ.2)JROTA=IRA
        CALL CWIND4(IVC(1,J),IVC(2,J),IWINS(J),FLOAT(JROTA),'C',
     +  QQQ,JCOOR)                          ! COMPUTE COORDINATES
        CALL DANDD(IDISP(1,J+3),IDISP(2,J+3),IXD(J),ESTAD(J),
     +  ENWPL(J),ITAREA,1,1,'A',CSDC,'C',JCOOR)
        JCF(J)=1-JCF(J)   ! IF FLAG=0 THEN SET TO 1 AND VV
120   CONTINUE
      IF(CSDC.EQ.'D')GOTO 80             ! GO REDRAW WINDOWS
C
130   WRITE(5,140)IRA
140   FORMAT(1X//' ROTATION ANGLE =',I4,' DEGREES.',/,
     + ' USE [-] AND [R] KEYS TO ROTATE AND PRESS [ENTER] TO ',
     + 'ACCEPT :',$)
      CALL KEY80(CVAL,COUT,JVAL)           ! GO READ KEYBOARD INPUT
      IF(CVAL.EQ.'E'.OR.CVAL.EQ.'Z')GOTO 160   ! ACCEPT ROTATION ANGLE.
      IF(COUT.EQ.'L'.OR.COUT.EQ.'R')THEN       ! LEFT/RIGHT ROTATION
        L=1
        IF(COUT.EQ.'L')L=-1                    ! LEFT ROTATION
        IRA=IRA+L*5                            ! EACH ROTATION STEP=5 DEG
        IF(IABS(IRA).LE.90)THEN                ! MAXIMUM ROTATION = 90 DEG
          JS=2
          JE=2
          GOTO 110                 ! REDRAW CROSS ON SEARCH IMAGE DISPLAY
         ELSE                      ! ANGLE GREATER THAN 90 DEG
          IRA=IRA-L*5
          GOTO 130
        END IF
       ELSE
        GOTO 130                   ! INVALID INPUT
      END IF
C
C     1.2 SET PARAMETERS TO DELETE CROSSES AND REDRAW THE WINDOWS.
C
160   JS=1
      JE=2
      CSDC='D'                     ! DELETE CROSS
      CSDW='S'                     ! DRAW WINDOW
      GOTO 110                     ! GO DELETE CROSSES AND DRAW WINDOWS
C
C     1.3 RETURN
C
500   CDV2='C'
      RETURN
```

```
PIP>TI:=MT2COM.FTN
C
C       SUBROUTINE MT2COM.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       MT2COM IS CALLED BY MTRAC2 TO READ WINDOW DATA, DO A ROTATION OF THE
C       TEMPLATE WINDOW DATA IF NECESSARY AND TO USE MAXIMUM CORRELATION
C       COEFFICIENTS TO DETERMINE THE VECTOR END POINT.
C
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE MT2COM(CDV1,CDV2,IVC,IDISP,IFEAT,IWINS,IXD,ESTAD,
     + ENWPL,LUNID,LUNDF,IVDN,SFAC,CPOS,IVNUM,IRNUM,ITAREA,JVECF,
     + JDOTF,JWINF,IRA,IVCOOR,IWCOOR)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IVC(2,2),IDISP(2,5),IFEAT(5,2),IWINS(2),IXD(2),
     + ESTAD(2),ENWPL(2),ITAREA(2,2),JDOTF(2),JWINF(2),IVCOOR(2,4),
     + IWCOOR(2,5,2),
     + ITARR(17,17),IARR(27,27),ITSET(2)
      CHARACTER*7 CMOD,CSUB
C
      IERR=1
      CSUB='MT2COM '
      ITSET(1)=1
      ITSET(2)=1
C
C       1.0 CHECK
C
      IF(CDV1.EQ.'C')THEN                    ! FUNCTION NOT FOR CURSOR MODE
        WRITE(5,40)
40      FORMAT(1X,//,' FUNCTION NOT APPLICABLE TO CURSOR MODE.',/)
        CDV2='A'
        GOTO 500                             ! RETURN
      END IF
C
      IF(JWINF(1).EQ.1.AND.JWINF(2).EQ.1)GOTO 80  ! WINDOWS MUST BE DEFINED
      WRITE(5,60)
60    FORMAT(1X//' SORRY! WINDOWS NOT DEFINED. CANNOT EXECUTE.',/)
      CDV2='D'
      GOTO 500                               ! RETURN
C
C       2.0 SET PARAMETERS TO DELETE WINDOWS
C
80    JS=1
      JE=2
      JQES=0
      JDRF=0
      CALL MT2VC2(CDV1,IVC,IDISP,IWINS,IXD,ESTAD,ENWPL,ITAREA,
     + JDOTF,JWINF,IRA,IWCOOR,JS,JE,JQES,JDRF,QQQ,IERR)
      IF(IERR.NE.1)GOTO 500
C
C       3.0 READ WINDOW DATA
C
90    DO 160 J=1,2                           ! 1=TEMPLATE, 2=SEARCH IMAGE
        IF(J.EQ.1.AND.IRA.NE.0)THEN          ! WINDOW WAS ROTATED
```

```
            CALL RTWIN(IVC(1,J),IVC(2,J),IFEAT(1,J),IFEAT(2,J),
     +      IFEAT(3,J),IFEAT(4,J),IFEAT(5,J),IWINS(J),LUNID,FLOAT(IRA),
     +      QQQ,IERR,ITARR)
            IF(IERR.NE.1)THEN
              CMOD='RTWIN  '
              GOTO 400                          ! RETURN WITH ERROR
            END IF
            GOTO 160
          END IF
C
          CALL RWIND(IVC(1,J),IVC(2,J),IFEAT(1,J),IFEAT(2,J),IFEAT(3,J),
     +    IFEAT(4,J),IFEAT(5,J),IWINS(J),LUNID,QQQ,IERR,IARR)
          IF(IERR.NE.1)THEN
           CMOD='RWIND  '
           GOTO 400                             ! RETURN WITH ERROR
          END IF
C
          IF(J.EQ.1)THEN
            DO 140 K=1,IWINS(1)
              DO 120 L=1,IWINS(1)
                 ITARR(L,K)=IARR(L,K)
120           CONTINUE
140         CONTINUE
          END IF
160     CONTINUE
C       WRITE(6,162)IWINS(1)
C162    FORMAT(1X//' ITARR :',I5)
C       DO 165 L=1,IWINS(1)
C         WRITE(6,170)((ITARR(L,K),K=1,IWINS(1)))
C165    CONTINUE
C       WRITE(6,167)IWINS(2)
C167    FORMAT(1X//' IARR : ',I5)
C       DO 175 L=1,IWINS(2)
C         WRITE(6,170)((IARR(L,K),K=1,IWINS(2)))
C170      FORMAT(1X,15I5)
C175    CONTINUE
C
C       4.0 COMPUTE CORRELATION COEFFICIENT AND DRAW VECTOR
C
        CALL CRCOEF(IWINS(1),IWINS(2),ITARR,IARR,QQQ,RMAXCC,L2,P2)
        IVC(1,2)=IVC(1,2)+L2-IWINS(2)/2-1
        IVC(2,2)=IVC(2,2)+P2-IWINS(2)/2-1
        WRITE(5,180)RMAXCC,IVC(1,2),IVC(2,2)
180     FORMAT(1X/' MAXIMUM CORRELATION COEFFICIENT =',F5.2,/,
     + ' AT LINE',I5,' PIXEL',I5)
C
          CFLD='A'
          CSD='S'
          CALL MT2AR2(IVC,IDISP,ITSET,IXD,ESTAD,ENWPL,LUNDF,IVDN,
     + SFAC,CPOS,IVNUM,IRNUM,ITAREA,JVECF,IVCOOR,CFLD,CSD)
          CDV2='A'
          GOTO 500
C
C       4.0 RETURN
C
400     WRITE(5,420)IERR,CMOD,CSUB
420     FORMAT(1X//' ERROR NO.',I5,' REPORTED BY ',A7,'SUBROUTINE ',
     + A7,//)
500     RETURN
        END
```

```
C
C       SUBROUTINE MT2AEX.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       MT2AEX IS CALLED BY MTRAC2 TO ACCEPT A VECTOR DRAWN IN ANNOTATION FIELD
C       ( THE VECTOR IS FIRST DRAWN IN ANNOTATION AND ONLY WHEN 'ACCEPTED'
C       BY THE USER IS IT REDRAWN AS A 'PERMANENT' VECTOR IN A THEME FIELD AND
C       THE COORDINATES STORED IN THE DATA FILE), TO DELETE A VECTOR ( ANNOATION
C       OR THEME FIELD) OR TO CLEAN UP THE DISPLAY BEFORE EXITTING.
C
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE MT2AEX(CDV1,CDV2,IVC,IDISP,IWIN5,ITSET,IXD,ESTAD,
     + ENWPL,LUNDF,IVDN,SFAC,CPOS,IVNUM,IRNUM,ITAREA,JVECF,JDOTF,
     + JWINF,IRA,IVCOOR,IWCOOR)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IVC(2,2),IDISP(2,5),IWIN5(2),ITSET(2),IXD(2),ESTAD(2),
     + ENWPL(2),ITAREA(2,2),JDOTF(2),JWINF(2),IVCOOR(2,4),IWCOOR(2,5,2)
C
      IERR=1
C
C       1.0 CHECK FOR AN ANNOTATION FIELD VECTOR. IF ONE EXIST, THEN DELETE IT.
C
      IF(JVECF.EQ.0)THEN                          ! NO VECTOR
         IF(CDV2.EQ.'A')THEN
           WRITE(5,40)
40           FORMAT(1X//' NO DEFINED VECTOR.  CANNOT EXECUTE.')
           CDV2='D'
           IF(CDV1.EQ.'W')CDV2='C'
           GOTO 500                               ! RETURN
         END IF
        ELSE                                      ! VECTOR EXIST. DELETE
         CFLD='A'
         CSD='D'
         IF(CDV2.EQ.'A')THEN                      ! SET TO REDRAW IN THEME FIELD
           CFLD='T'
           CSD='S'
         END IF
C
         CALL MT2AR3(IVC,IDISP,ITSET,IXD,ESTAD,ENWPL,LUNDF,IVDN,SFAC,
     +     CPOS,IVNUM,IRNUM,ITAREA,JVECF,IVCOOR,CFLD,CSD)
         IF(CDV2.EQ.'E')THEN
           CDV2='D'
           GOTO 500                               ! RETURN
         END IF
       END IF
C
      IF(CDV2.EQ.'E')GOTO 100                     ! ERASE VECTOR
      IF(CDV2.EQ.'X')GOTO 200                     ! CLEAN UP BEFORE EXIT.
C
C       2.0 ACCEPT VECTOR DRAWN IN ANNOTATION FIELD. ALL WORK DONE ALREADY.
C           SET DEFAULT CDV2 AND RETURN.
C
      CDV2='D'
```

```
      GOTO 500
C
C     3.0 ERASE A VECTOR IN THEME FIELD.
C
100   JS=IVDN                          ! SET PARAMETERS FOR USER TO IDENTIFY
      JE=JS                            ! THE START OR END POINT OF THE ARROW
      JQES=1                           ! (VECTOR) TO BE REMOVED. IF VECTORS
      JDRF=0                           ! DISPLAYED ON TEMPLATE IMAGE, IT WILL
                                       ! BE THE START POINT, ELSE THE END POINT
      CALL MT2VC3(CDV1,IVC,IDISP,IWINS,IXD,ESTADR,ENWPL,ITAREA,JDOTF,
     + JWINF,IRA,IWCOOR,JS,JE,JQES,JDRF,QQQ,IERR)
      IF(IERR.NE.1)GOTO 500            ! RETURN
C
      CFLD='T'
      CSD='D'
      CALL MT2AR3(IVC,IDISP,ITSET,IXD,ESTAD,ENWPL,LUNDF,IVDN,SFAC,
     + CPOS,IVNUM,IRNUM,ITAREA,JVECF,IVCOOR,CFLD,CSD)
      GOTO 500
C
C     4.0 EXIT. FIRST ERASE ALL EXISTING ANNOTATION ( DOTS AND WINDOWS -
C         ARROWS WOULD HAVE ALREADY BEEN REMOVED BY 1.0 ABOVE)
C
200   IF(JDOTF(1).NE.0.OR.JDOTF(2).NE.0)GOTO 220     ! MUST DELETE DOTS
      IF(JWINF(1).NE.1.OR.JWINF(2).NE.0)GOTO 220     ! MUST DELETE WINDOWS
      GOTO 500
C
220   JS=1
      JE=2
      JQES=0
      JDRF=0
      CALL MT2VC3(CDV1,IVC,IDISP,IWINS,IXD,ESTADR,ENWPL,ITAREA,JDOTF,
     + JWINF,IRA,IWCOOR,JS,JE,JQES,JDRF,QQQ,IERR)
C
C     5.0 RETURN
C
500   RETURN
      END
```

```
C
C         SUBROUTINE MT2VEC.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       THIS SUBROUTINE IS CALLED BY MT2DMS,MT2COM AND MT2AEX. IT MARKS THE
C       VECTOR START AND END POINTS WITH A DOT, DRAW A WINDOW OR DELETE THESE
C       MARKS.
C
C       IN ORDER TO BE ABLE TO USE THE SUBROUTINE IN 3 DIFFERENT OVERLAYS
C       3 DIFFERENTLY NAMED COPIES OF THE SAME ROUTINE MUST EXIST : THE ARE
C       CALLED AS MT2VC1, MT2VC2 AND MT2VC3 RESPECTIVELY
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE MT2VC1(CDV1,IVC,IDISP,IWINS,IXD,ESTAD,ENWPL,
     + ITAREA,JDOTF,JWINF,IRA,IWCOOR,JS,JE,JQES,JDRF,QQQ,IERR)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IVC(2,2),IDISP(2,5),IWINS(2),IXD(2),ESTAD(2),
     + ENWPL(2),ITAREA(2,2),JDOTF(2),JWINF(2),IWCOOR(2,5,2),
     + JCOOR(2,5)
      CHARACTER CMOD*7,CSUB*7,CPOINT(2)*5
C
      DATA CPOINT/'START','END  '/
      IERR=1
      CSUB='MT2VC1 '
C
C       1.0 LINK TO APPROPRIATE DISPLAY AND MOVE CURSOR.
C
      J=JS
20    CALL ZMDISP(J,IDISP(1,J),-1,QQQ,IERR)      ! LINK TO DISPLAY
      IF(IERR.NE.1)THEN
        CMOD='ZMDISP '
        GOTO 200                                 ! RETURN WITH ERROR
      END IF
C
      CALL ZMMCUR(IVC(1,J),IVC(2,J),0,QQQ,IERR) ! MOVE CURSOR
      IF(IERR.NE.1)THEN
        CMOD='ZMMCUR '
        GOTO 200                                 ! RETURN WITH ERROR.
      END IF
C
C       2.0 CLEAN UP THE DISPLAY.
C
C       2.1 REMOVE DOT MARKER IF IT EXISTS.
C
      IF(JDOTF(J).EQ.1)THEN                      ! YES
       JCOOR(1,1)=IVC(1,J)
       JCOOR(2,1)=IVC(2,J)
       CALL ZMMCUR(IVC(1,J)+1,IVC(2,J)+1,0,QQQ,IERR)
       CALL DANDD(IDISP(1,J+3),IDISP(2,J+3),IXD(J),ESTAD(J),
     +  ENWPL(J),ITAREA,1,1,'A','D','D',JCOOR)
       JDOTF(J)=0                                ! FLAG TO SHOW DOT REMOVED
      END IF
C
C       2.2 REMOVE WINDOW IF IT EXISTS.
```

```
C
      IF(JWINF(J).EQ.1)THEN                    ! YES
       DO 24 K=1,5
         JCOOR(1,K)=IWCOOR(1,K,J)
         JCOOR(2,K)=IWCOOR(2,K,J)
24     CONTINUE
       CALL DANDD(IDISP(1,J+3),IDISP(2,J+3),IXD(J),ESTAD(J),
     +  ENWPL(J),ITAREA,1,1,'A','D','W',JCOOR)
       JWINF(J)=0                              ! FLAG TO SHOW WINDOW REMOVED.
      END IF
C
C     3.0 DEFINE VECTOR START AND END POINTS.
C
      IF(JQES.EQ.1)THEN                        ! FLAG REQUESTING USER INTERACTION
30      WRITE(5,40)CPOINT(J)
40      FORMAT(1X/' MOVE CURSOR TO VECTOR ',A5,' POINT PLEASE.',/,
     +  ' PRESS CENTER] WHEN READY OR [CTRL-Z] TO EXIT :',$)
        CALL KEYBD(CVAL,COUT,JVAL)             ! READ KEY INPUT
        IF(CVAL.EQ.'Z')GOTO 300                ! CTRL-Z .  RETURN
        IF(CVAL.NE.'E')GOTO 30                 ! INVALID ENTRY
C
        CALL ZMCUSR(QQQ,IERR,JDUM,IVC(1,J),IVC(2,J))   ! READ CURSOR POSITION
        IF(IERR.NE.1)THEN
          CMOD='ZMCUSR '
          GOTO 200                             ! RETURN WITH ERROR
        END IF
      END IF
C
C     3.1 PLACE DOT MARKER IF REQUESTED (JDRF=1)
C
      IF(JDRF.EQ.1)THEN                        ! YES, PLACE NEW MARK.
        JCOOR(1,1)=IVC(1,J)
        JCOOR(2,1)=IVC(2,J)
        CALL ZMMCUR(IVC(1,J)+1,IVC(2,J)+1,0,QQQ,IERR)  ! MOVE CURSOR
        CALL DANDD(IDISP(1,J+3),IDISP(2,J+3),IXD(J),ESTAD(J),ENWPL(J),
     +  ITAREA,1,1,'A','S','D',JCOOR)
        JDOTF(J)=1                             ! SET FLAG TO SHOW MARK WAS REPLACED
      END IF
C
C     3.2 DRAW WINDOW IF REQUIRED ( JDRF=1 )
C
      IF(CDV1.EQ.'W')THEN                      ! ONLY APPLY TO WINDOW MODE
80      IF(JDRF.EQ.1)THEN                      ! FLAG REQUEST NEW WINDOW TO BE DRAWN
          ROTA=0.0
          IF(J.EQ.2)ROTA=FLOAT(IRA)
          CALL CWIND1(IVC(1,J),IVC(2,J),IWINS(J),ROTA,'W',
     +    QQQ,JCOOR)                           ! COMPUTE COORDINATES
          CALL DANDD(IDISP(1,J+3),IDISP(2,J+3),IXD(J),ESTAD(J),
     +    ENWPL(J),ITAREA,1,1,'A','S','W',JCOOR)   ! DRAW THE WINDOW
          DO 90 K=1,5
            IWCOOR(1,K,J)=JCOOR(1,K)           ! STORE NEW COORDINATES
            IWCOOR(2,K,J)=JCOOR(2,K)
90        CONTINUE
          JWINF(J)=1
        END IF
      END IF
C
      J=J+1
      IF(J.LE.JE)GOTO 20                       ! DO NEXT DISPLAY
      GOTO 300                                 ! DONE.  RETURN
```

```
C
C       5.0 RETURN
C
200     WRITE(5,220)IERR,CMOD,CSUB
220     FORMAT(1X//' ERROR NO.',I5,' REPORTED BY ',A7,'SUBROUTINE ',
     + A7,//)
300     RETURN
        END
```

```
C
C       MT2ARR.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       THIS SUBROUTINE IS CALLED BY MT2DMS, MT2COM AND MT2AEX TO DRAW OR
C       DELETE THE ARROWS REPRESENTING VECTORS. VECTOR COORDINATES ARE
C       COMPUTED BY SUBROUTINE CARROW AND DRAWING ON THE SCREEN DONE BY
C       SUBROUTINE DANDD ( DRAW-AND-DELETE)
C
C       NB ! IN ORDER TO INTRODUCE MT2ARR IN DIFFERENT BRANCHES OF THE
C            MTRACK OVERLAY STRUCTURE 3 DIFFERENT COPIES THERE-OF MUST
C            EXIST AND ARE NAMED MT2AR1, MT2AR2 AND MT2AR3
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE MT2AR1(IVC,IDISP,ITSET,IXD,ESTAD,ENWPL,LUNDF,
     + IVDN,SFAC,CPOS,IVNUM,IRNUM,ITAREA,JVECF,IVCOOR,CFLD,CSD)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IVC(2,2),IDISP(2,5),ITSET(2),IXD(2),ESTAD(2),ENWPL(2),
     + ITAREA(2,2),IVCOOR(2,4),
     + JCOOR(2,5),JVC(2,2)
C
C
      J=IVDN                          ! DISPLAY USED FOR VECTOR ANNOTATION
      IF(JVECF.EQ.1)THEN              ! IS THERE AN ANNOTATION FIELD VECTOR ?
       DO 20 K=1,4                    ! YES. DELETE IT.
         JCOOR(1,K)=IVCOOR(1,K)       ! IVCOOR() STORE COORDINATES FOR
         JCOOR(2,K)=IVCOOR(2,K)       ! MOST RECENTLY PRODUCED ARROW(VECTOR)
20     CONTINUE
       CALL DANDD(IDISP(1,J+3),IDISP(2,J+3),IXD(J),ESTAD(J),ENWPL(J),
     +  ITAREA,1,1,'A','D','V',JCOOR)
       JVECF=0                        ! SET FLAG TO 0 = NO DEFINED VECTOR
      END IF
C
      IF(CFLD.EQ.'A'.AND.CSD.EQ.'D')GOTO 200 ! DONE.  RETURN
C
C     COMPUTE COORDINATES FOR A NEW VECTOR ?
C
      IF((CFLD.EQ.'T'.AND.CSD.EQ.'D').OR.(CFLD.EQ.'A'.AND.CSD.EQ.'S'))
     + THEN      ! YES
        IF(CFLD.EQ.'T')THEN                ! DELETE A THEME VECTOR. FIND IT
          DO 40 K=1,IRNUM                  ! IN THE STORAGE FILE. READ START/
            READ(LUNDF,REC=K,FMT=30)N,N1,N2,N3,N4,JVC  ! END POINT COORDINATE
30          FORMAT(1X,9I6)
            IF(JVC(1,J).EQ.IVC(1,J).AND.JVC(2,J).EQ.IVC(2,J).AND.
     +      N.NE.0)THEN
             IVC(1,1)=N1                   ! RECOVER ORIGINAL CURSOR POSITIONS
             IVC(2,1)=N2
             IVC(1,2)=N3
             IVC(2,2)=N4
             N=0
             WRITE(LUNDF,REC=K,FMT=30)N,N1,N2,N3,N4,JVC  ! ERASE BY N=0
             IVNUM=IVNUM-1               ! DECREMENT NUMBER OF DEFINED VECTORS
             GOTO 60
            END IF
```

```
40          CONTINUE
            WRITE(5,50)
50          FORMAT(1X//' VECTOR COULD NOT BE LOCATED IN STORAGE FILE - ',
     +      'NOT DELETED.'/)
            GOTO 200
          END IF
C
60        CALL CARRO1(IVC,SFAC,CPOS,QQQ,JCOOR)   ! COMPUTE VECTOR COORDINATES
          DO 80 K=1,4
            IVCOOR(1,K)=JCOOR(1,K)
            IVCOOR(2,K)=JCOOR(2,K)
80        CONTINUE
        END IF
C
        CALL DANDD(IDISP(1,J+3),IDISP(2,J+3),IXD(J),ESTAD(J),ENWPL(J),
     + ITAREA,ITSET(1),ITSET(2),CFLD,CSD,'V',JCOOR)    ! DRAW/DELETE VECTOR
        IF(CFLD.EQ.'A'.AND.CSD.EQ.'S')JVECF=1               ! SET FLAG
C
C
        IF(CFLD.EQ.'A')GOTO 200          ! DONE. RETURN
        IF(CSD.EQ.'S')THEN               ! NEW VECTOR CREATED (THEME FIELD) MUST
          IVNUM=IVNUM+1                  ! WRITE TO STORAGE FILE.
          IRNUM=IRNUM+1           ! RECORD NUMBER IN STORAGE FILE
          WRITE(LUNDF,REC=IRNUM,FMT=90)IRNUM,IVC,
     +    (((IVCOOR(K,L)),K=1,2),L=1,4)
        END IF
C
C       RETURN
C
200     RETURN
        END
```

```
C
C       SUBROUTINE CWINDO.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       THIS SUBROUTINE IS CALLED INDIRECTLY ( VIA MT2VEC ) BY MT2DMS,MT2COM,
C       AND MT2AEX AND DIRECTLY BY MT2ROT. TO BE ABLE TO INCLUDE IT IN 4
C       BRANCHES OF THE OVERLAY STRUCTURE 4 COPIES OF IT EXIST AND ARE
C       CALLED RESPECTIVELY AS CWIND1,CWIND2,CWIND3 AND CWIND4.
C
C       CWINDO COMPUTES THE CORNER COORDINATES FOR A WINDOW OR THE END POINTS
C       OF THE TWO LINE SEGMENTS FOR A CROSS.
C
C       CWINDO(JCL,JCP,SROT,CWC,QQQ,JCOOR)
C       JCL,JCP          INT*2  WINDOW/CROSS CENTRE LINE AND PIXEL
C       JWD                ,,   WINDOW/CROSS DIMENSIONS. NOTE ! THE WINDOW FRAME
C                               WILL BE COMPUTED SUCH AS NOT TO COVER ANY OF THE
C                               ACTUAL WINDOW DATA AREA IE ONE LINE TO TOP OR
C                               BOTTOM ETC. THE CROSS WILL BE THE ACTUAL SPECIFI
C                               DIMENSION.
C       SROT             REAL*4 ROTATION ANGLE (DEG. + = CLOCKWISE)
C       CWC              CHAR*1 'W'=WINDOW,  'C'=CROSS
C       JCOOR(2,5)       INT*2  COORDINATES. (1,)=LINE, (2,)=PIXEL. IF CWC='W':
C                               (,1)&(,5)=TOP LEFT, (,2)=TOP RIGHT, (,3)=BOTTOM
C                               RIGHT, (,4)=BOTTOM LEFT CORNER. IF CWC='C' :
C                               (,1)=TOP, (,2)=BOTTOM, (,3)=LEFT, (,4)=RIGHT
C
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE CWIND1(JCL,JCP,JWD,SROT,CWC,QQQ,JCOOR)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION JCOOR(2,5),JWC(2,4,2),JCLP(2)
      DATA JWC/-1,-1,-1,1,1,1,1,-1,-1,0,1,0,0,-1,0,1/
C
      RPI=4.0*ATAN(1.0)                          ! PI
      RDR=RPI/180.0                              ! DEGREE TO RADIAN CONVERSION
      SRAD=RDR*SROT                              ! ROTATION ANGLE IN RADIANS
      JCLP(1)=JCL
      JCLP(2)=JCP
      J=1
      IF(CWC.EQ.'C')J=2
      JH=JWD/2
      IF(CWC.EQ.'W')JH=JH+1
C
C       1.0 SET UP THE INITIAL CORNER/END POINT COORDINATES
C
      DO 40 K=1,4
        DO 20 L=1,2                    ! 1=LINE, 2=PIXEL
          JCOOR(L,K)=JCLP(L)+JWC(L,K,J)*JH
20      CONTINUE
40    CONTINUE
C
C       2.0 ROTATE
C
      IF(SROT.EQ.0.0)GOTO 60
```

```
      RCL=FLOAT(JCL)
      RCP=FLOAT(JCP)
      DO 50 J=1,4
        SY=FLOAT(JCL-JCOOR(1,J))
        SX=FLOAT(JCOOR(2,J)-JCP)
        SR=SQRT(SY*SY+SX*SX)
        STH=ATAN2(SX,SY)
        IF(STH.LT.0.0)STH=STH+2.0*RPI
        STH=STH+SRAD
        JCOOR(1,J)=INT(RCL-SR*COS(STH))
        JCOOR(2,J)=INT(RCP+SR*SIN(STH))
50    CONTINUE
C
C     9.0 RETURN
C
60    IF(CWC.EQ.'W')THEN
       JCOOR(1,5)=JCOOR(1,1)
       JCOOR(2,5)=JCOOR(2,1)
      END IF
C
      RETURN
      END
```

```
C
C       SUBROUTINE CARROW.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING
C
C       THIS SUBROUTINE IS CALLED INDIRECTLY ( VIA MT2ARR ) BY MT2DMS,
C       MT2COM AND MT2AEX. IN ORDER TO BE ABLE TO INCLUDE IT IN 3 BRANCHES
C       OF THE OVERLAY STRUCTURE 3 COPIES OF CARROW MUST EXIST AND ARE
C       CALLED AS CARRO1, CARRO2 AND CARRO3 RESPECTIVELY.
C
C       CARROW COMPUTES THE COORDINATES OF AN ARROW REPRESENTING AN ADVECTION
C       VECTOR. THE ARROW HAS A FEATHER ON THE TAIL. THE SIZE (LENGTH) OF THE
C       ARROW IS DETERMINED BY A SCALING FACTOR (SFAC EG. SFAC=2 THEN THE
C       ARROW WILL BE TWICE THE LENGTH OF THE DISTANCE BETWEEN THE SUPPLIED
C       START AND END POINTS, SFAC=0.5 AND IT WILL BE HALF THE DISTANCE, ETC)
C       THE ARROW MAY BE DRAWN STARTING AT THE GIVEN START POINT , TOWARDS
C       THE GIVEN END POINT (CPOS='S') OR DRAWN TO BE CENTRED ON THE GIVEN
C       START POINT (CPOS='C')
C
C       CARROW(IVC,SFAC,CPOS,QQQ,JCOOR)
C       IVC(2,2)        INT*2   (1,)=LINE, (2,)=PIXEL. (,1)=START, (,2)=END POINT
C       SFAC            REAL*4  VECTOR SCALING FACTOR
C       CPOS            CHAR*1  'S'= ARROW START ON IVC(1,1)/IVC(2,1)
C                               'C'= ARROW CENTRED ON IVC(1,1)/IVC(2,1)
C       JCOOR(2,5)      INT*2   COORDINATES. (1,)&(2,)=LINE AND PIXEL. (,1)=START
C                               POINT (FEATHER SIDE), (,2)=END POINT, (,3)&(,4)=
C                               START AND END POINTS OF THE FEATHER.
C
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE CARRO1(IVC,SFAC,CPOS,QQQ,JCOOR)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IVC(2,2),JCOOR(2,5),SARD(8),IFEATH(4,8)
C
      DATA SARD/22.5,67.5,112.5,157.5,202.5,247.5,292.5,317.5/
      DATA IFEATH/0,-1,0,1,-1,-1,1,1,-1,0,1,0,-1,1,1,-1,0,-1,0,1,
     + -1,-1,1,1,-1,0,1,0,1,-1,-1,1/
C
      RPI=4.0*ATAN(1.0)                 ! PI
      RDR=RPI/180.0                     ! DEGREES TO RADIANS CONVERSION
C
C       1.0 COMPUTE SIZE AND DIRECTION OF THE ARROW FROM IVC()
C
      SX=FLOAT(IVC(2,2)-IVC(2,1))
      SY=FLOAT(IVC(1,1)-IVC(1,2))
      SR=SQRT(SX*SX+SY*SY)
      SRSC=SR*SFAC                      ! SIZE AFTER SCALING
      IF(SRSC.LT.1.0)GOTO 400           ! TOO SMALL. DRAW VECTOR AS A '+'
      STHR=ASIN(SX/SR)                  ! DIRECTION IN RADIANS
      STH=STHR/RDR                      !      ,,     IN DEGREES
C
      IF(SY.GE.0.0)THEN
        STH=360+STH
```

```
        ELSE
          STH=180-STH
        END IF
        IF(STH.GE.360.0)STH=STH-360
C
C       2.0 WHICH TYPE OF FEATHER ?  (DEPENDS UPON ARROW DIRECTION)
C
        DO 100 JCL=1,7
          IF(STH.GT.SARD(JCL).AND.STH.LE.SARD(JCL+1))GOTO 120
100     CONTINUE
        JCL=0
120     JCL=JCL+1          ! TYPE OF FEATHER
C
C       3.0 IF NECESSARY ( DUE TO SFAC OR CPOS ) COMPUTE NEW COORDINATES FOR TH
C           ARROW SHAFT.
C
        DO 220 J2=1,2
          DO 200 J1=1,2
            JCOOR(J1,J2)=IVC(J1,J2)
200       CONTINUE
220     CONTINUE
        IF(SFAC.EQ.1.0.AND.CPOS.EQ.'S')GOTO 300     ! NO ACTION REQUIRED
        IF(CPOS.EQ.'S')THEN                         ! COMPUTE NEW END POINT ONLY
          JCOOR(1,2)=IINT(FLOAT(IVC(1,1))-SFAC*SY+0.5)
          JCOOR(2,2)=IINT(FLOAT(IVC(2,1))+SFAC*SX+0.5)
         ELSE
          JCOOR(1,2)=IINT(FLOAT(IVC(1,1))-SFAC*SY*0.5+0.5)    ! ARROW END POINT
          JCOOR(2,2)=IINT(FLOAT(IVC(2,1))+SFAC*SX*0.5+0.5)    ! COORDINATES
          JCOOR(1,1)=IINT(FLOAT(IVC(1,1))+SFAC*SY*0.5+0.5)    !ARROW START POINT
          JCOOR(2,1)=IINT(FLOAT(IVC(2,1))-SFAC*SX*0.5+0.5)    ! COORDINATES
        END IF
C
C       4.0 ADD COORDINATES FOR THE FEATHER
C
300     JCOOR(1,3)=JCOOR(1,1)+IFEATH(1,JCL)
        JCOOR(2,3)=JCOOR(2,1)+IFEATH(2,JCL)
        JCOOR(1,4)=JCOOR(1,1)+IFEATH(3,JCL)
        JCOOR(2,4)=JCOOR(2,1)+IFEATH(4,JCL)
        GOTO 500
C
C       5.0 DEAL WITH A ZERO VECTOR. DRAW AS A '+'
C
400     JCOOR(1,1)=IVC(1,1)
        JCOOR(2,1)=IVC(2,1)+1
        JCOOR(1,2)=IVC(1,1)
        JCOOR(2,2)=IVC(2,1)-1
        JCOOR(1,3)=IVC(1,1)-1
        JCOOR(2,3)=IVC(2,1)
        JCOOR(1,4)=IVC(1,1)+1
        JCOOR(2,4)=IVC(2,1)
C
C       6.0 RETURN
C
500     RETURN
        END
```

```
C
C       SUBROUTINE DANDD.FTN        - ( DRAW-AND-DELETE)
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       THIS SUBROUTINE IS CALLED TO PRODUCE ON THE VIDEO DISPLAY THE VARIOUS
C       STRUCTURES REQUIRED FOR MTRAC2 IE. DOTS TO MARK THE START AND END
C       POINTS OF DEFINED VECTORS, WINDOWS (SQUARES), CROSSES AND ARROWS.
C       ARROWS REPRESENTING THE ADVECTION VECTORS MAY BE DRAWN IN THE ANNO-
C       TATION OR THEME FIELD. ALL OTHERS ARE DRAWN ONLY IN ANNOTATION FIELD
C       BY MTRAC2.
C       DANDD IS ALSO USED TO DELETE PREVIOUSLY DRAWN STRUCTURES.
C       WHEN USING THE THEME FIELD, RECORD IS KEPT ABOUT THE AFFECTED AREA
C       OF THE DISPLAYED IMAGE - IE. THE AREA WHICH NEEDS TO BE SAVED ON DISK.
C
C       DANDD(JSL,JSP,JIXD,ESTADR,EWPL,ITAREA,ITMSK,ITVAL,CFLD,CSD,CFUNC,
C             JCOOR)
C
C       JSL,JSP          INT*2   START LINE AND PIXEL OF THE DISPLAY
C       JIXD               ,,    IXEL DEPTH ( NO. OF BITS USED TO DISPLAY 1 PIXEL)
C       ESTADR           INT*4   START ADDRES OF THE DISPLAY IN VIDEO MEMORY
C       EWPL               ,,    NUMBER OF WORDS USED TO DISPLAY ONE LINE
C       ITAREA(2,2)        ,,    AREA OF IMAGE AFFECTED BY THEME MODIFICATIONS
C                                (1,)&(2,)=LINE AND PIXEL; (,1)=TOP LEFT CORNER
C                                (,2)=BOTTOM RIGHT CORNER
C       ITMSK,ITVAL      INT*2   BIT MASK AND VALUE NEEDED TO SET/DELETE THEME
C       CFLD             CHAR*1  'A'=ANNOTATION FIELD, 'T'=THEME FIELD
C       CSD                ,,    'S'=SET THE FIELD,    'D'=DELETE THE FIELD
C       CFUNC              ,,    'D'=DOT ; 'W'=WINDOW ; 'C'=CROSS ; 'V'=VECTOR
C       JCOOR(2,5)       INT*2   COORDINATES FOR LINE SEGMENTS. (1,) & (2,)= LINE
C                                PIXEL. IN CASE OF A WINDOW THE SEGMENTS ARE :
C                                (,1)-(,2), (,2)-(,3) ETC. IN CASE OF A CROSS OR
C                                VECTOR THE TWO SEGMENTS ARE (,1)-(2) AND (,3)-(,4
C                                AND IN CASE OF A DOT USE (,1)
C
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE DANDD(JSL,JSP,JIXD,ESTADR,EWPL,ITAREA,ITMSK,ITVAL,
     + CFLD,CSD,CFUNC,JCOOR)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-Z)
      DIMENSION ITAREA(2,2),JCOOR(2,5),BVMA(2),BY(2),JLP(2,2)
      CHARACTER CMOD*7,CSUB*5
      EQUIVALENCE (BVMA(1),JRVMA),(BY(1),JEQ)
C
C       1.0 INITIALISE.
C
      IERR=1
      CSUB='DANDD'
      JINDEX=5
      IF(JIXD.EQ.7)JINDEX=4
      IF(CFLD.EQ.'A')THEN          ! SET MASK AND VALUE FOR ANNOTATION BIT
        EMSK=1
        EVAL=1
        IF(CSD.EQ.'D')EVAL=0       ! TO DELETE
      END IF
```

```
C
C       WRITE(6,10)JSL,JSP,JIXD,ESTADR,EWPL,ITMSK,ITVAL,
C     + CFLD,CSD,CFUNC,JCOOR
C10     FORMAT(1X//' DANDO (10) :',/,
C     + ' JSL,JSP,JIXD=',3I5,'  ESTADR,EWPL=',I8,I4,/,
C     + ' ITMSK,ITVAL=',2I3,'  CFLD,CSD,CFUNC=',3(1X,A1),/,
C     + ' JCOOR=',5(I4,I5,3X),/)
C
C       2.0 SET/DELETE A DOT IN ANNOATION FIELD
C
        IF(CFUNC.EQ.'D')THEN
          EREG=ESTADR+(JCOOR(1,1)-JSL)*2*EWPL
          JPIX=JCOOR(2,1)-JSP+1
          GOTO 400                    ! GO SET/DELETE THE DOT AND RETURN.
        END IF
C
C       3.0 DRAW/DELETE A WINDOW IN ANNOTATION FIELD
C
        IF(CFUNC.EQ.'W')THEN
          ASSIGN 80 TO JGO
          DO 100 L=1,4                ! THE 4 SIDES
            DO 40 K=1,2               ! 1=LINE; 2=PIXEL
              JLP(K,1)=JCOOR(K,L)     ! FIRST POSITION
              JLP(K,2)=JCOOR(K,L+1)   ! SECOND POSITION
40          CONTINUE
            GOTO 200                  ! GO DRAW THE LINE
80          CONTINUE
100       CONTINUE
          GOTO 500                    ! RETURN
        END IF
C
C       4.0 DRAW/DELETE A CROSS OR AN ARROW
C
        IF(CFUNC.EQ.'C'.OR.CFUNC.EQ.'V')THEN
          ASSIGN 140 TO JGO
          DO 160 L=1,3,2              ! THE TWO SEGMENTS
            DO 120 K=1,2              ! 1=LINE, 2=PIXEL
              JLP(K,1)=JCOOR(K,L)     ! FIRST POSITION
              JLP(K,2)=JCOOR(K,L+1)   ! SECOND POSITION
120         CONTINUE
            GOTO 200                  ! GO DRAW THE LINE
140         CONTINUE
160       CONTINUE
          GOTO 500                    ! RETURN
        END IF
C
C ------------------------------------------------------------------------------
C
C       5.0 DRAW/DELETE LINES
C
C       5.1 SET UP THE INTERPOLATION EQUATION. IF LINE DIFFERENCE LARGER
C           THAN PIXEL DIFFERENCE THEN USE LINES AS X-VARIABLE, ELSE USE PIXELS
C
200     IF(IIABS(JLP(1,2)-JLP(1,1)).GE.IIABS(JLP(2,2)-JLP(2,1)))THEN
           J1=1                       ! LINES
           J2=2                       ! PIXELS
          ELSE
           J1=2                       ! PIXELS
           J2=1                       ! LINES
        END IF
```

```
      RA=FLOAT(JLP(J2,2)-JLP(J2,1))/FLOAT(JLP(J1,2)-JLP(J1,1))
      RB=FLOAT(JLP(J2,1))-FLOAT(JLP(J1,1))*RA
C
C     5.2 DO THE INTERPOLATION AND SET/DELETE THE PIXEL VALUES
C
      JINC=(JLP(J1,2)-JLP(J1,1))/IIABS(JLP(J1,2)-JLP(J1,1))
      JCCTLO=1
      JCCTPO=1
      DO 240 J=JLP(J1,1),JLP(J1,2),JINC
        K=INT(RB+FLOAT(J)*RA+0.45)
        JCCTL=J
        JCCTP=K
        IF(J1.EQ.2)THEN
         JCCTL=K
         JCCTP=J
        END IF
        IF(JCCTL.EQ.JCCTLO.AND.JCCTP.EQ.JCCTPO)GOTO 240
        GOTO 300                        ! SET/DELETE PIXEL FIELD
230     JCCTLO=JCCTL
        JCCTPO=JCCTP
240   CONTINUE
      GOTO JGO
C
C ------------------------------------------------------------------------------
C
C     6.0 SET OR DELETE A PIXEL FIELD.
C
300   EREG=ESTADR+(JCCTL-JSL)*2*EWPL
      JPDIF=JCCTP-JSP
      JPIX=JPDIF+1
      IF(CFLD.EQ.'A')THEN             ! SET/DELETE ANNOTATION FIELD
        CALL ZMTAWL(JPIX,JPIX,EREG,JINDEX,EMSK,EVAL)
        GOTO 230
      END IF
C
C     6.1 SET OR DELETE A THEME FIELD
C
      EMSK=ITMSK
      EVADR=EREG+JPDIF*2                          ! WORD ADDRES, 15-BIT IXEL
      IF(JIXD.EQ.7)EVADR=EREG+2*INT(FLOAT(JPDIF)/2.0)   ! 7-BIT IXEL
      CALL ZMRVMA(EVADR,1,QQQ,IERR,JRVMA)         ! READ A WORD FROM VMA
      IF(IERR.NE.1)THEN
        CMOD='ZMRVMA '
        WRITE(5,320)IERR,CMOD,CSUB
320     FORMAT(1X//' ERROR NO.',I5,' REPORTED BY ',A7,'SUBROUTINE ',
     +  A7,//)
      END IF
C
      IF(JIXD.EQ.7)THEN           ! 7-BIT IXEL. EXTRACT THE CORRECT BYTE FROM
         JEQ=0                    ! THE WORD
         IF(IMOD(JPDIF,2).EQ.0)THEN
            BY(1)=BVMA(2)         ! EVEN NO. OF PIXELS FROM IMAGE EDGE. USE #2
           ELSE
            BY(1)=BVMA(1)         ! ODD NUMBER. USE BYTE NUMBER 1
         END IF
         JRVMA=JEQ
      END IF
C
      JTV=IAND(JRVMA,ITMSK)       ! THEME CONTENT OF THE PIXEL
      IF(CSD.EQ.'S')THEN
```

```
          EVAL=IOR(JTV,ITVAL)    ! EVAL FOR SETTING THE THEME.
         ELSE
          EVAL=IAND(JTV,NOT(ITVAL)) ! EVAL FOR DELETING THE THEME
          IF(JTV.LT.ITVAL)EVAL=JTV  ! TO AVOID DELETING SMALLER NO. THEMES
        END IF
C
400     CALL ZMTAWL(JPIX,JPIX,EREG,JINDEX,EMSK,EVAL)
        IF(CFUNC.EQ.'D')GOTO 500
        ITAREA(1,1)=IMIN0(ITAREA(1,1),JCCTL)          ! MINIMUM LINE
        ITAREA(2,1)=IMIN0(ITAREA(2,1),JCCTP)          !    ,,     PIXEL
        ITAREA(1,2)=IMAX0(ITAREA(1,2),JCCTL)          ! MAXIMUM LINE
        ITAREA(2,2)=IMAX0(ITAREA(2,2),JCCTP)          !    ,,     PIXEL
        GOTO 290
C
C ------------------------------------------------------------------------------
C
C       7.0 RETURN
C
500     RETURN
        END
```

```
C
C       SUBROUTINE RWIND.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       RWIND IS CALLED BY MT2COM TO READ TEMPLATE OR SEARCH WINDOW DATA
C       FROM DISK FILES PRIOR TO COMPUTATION OF THE MAXIMUM CORRELATION
C       COEFFICIENT.
C
C       RWIND(JCL,JCP,JBH,JBL,JNP,JFL,JFP,JWD,LUNID,QQQ,IERR,IARR)
C       JCL,JCP     INT*2   WINDOW CENTRE LINE AND PIXEL
C       JBH,JBL       ,,    HIGH AND LOW ORDER BYTES OF THE BASE ADDRES FOR
C                           THE TEMPLATE IMAGE DISK FILE.
C       JNP           ,,    NUMBER OF PIXELS PER LINE IN THE TEMPLATE FILE.
C       JFL,JFP       ,,    FIRST LINE AND PIXEL IN TEMPLATE IMAGE FILE.
C       JWD           ,,    WINDOW DIMENSION
C       LUNID         ,,    LOGICAL UNIT NUMBER FOR THE IMAGE DISK
C       IERR          ,,    ERROR CODE. 1=NO ERROR
C       IARR(27,27)   ,,    ARRAY CONTAINING THE WINDOW DATA.
C
C
C       J.J.ASENSIO  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE RWIND(JCL,JCP,JBH,JBL,JNP,JFL,JFP,JWD,LUNID,
     + QQQ,IERR,IARR)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IARR(27,27),BUF(1050),BDUM(2)
      EQUIVALENCE (BDUM(1),JDUM)
C
      IERR=1
      JL1=JCL-JWD/2                     ! FIRST LINE TO READ
      JL2=JCL+JWD/2                     ! LAST  LINE TO READ
      JP1=JCP-JWD/2                     ! FIRST PIXEL TO READ
      JRELP=JP1-JFP+1                   ! PIXEL RELATIVE TO BEGINNING OF LINE
      JRELL=JL1-JFL                     ! LINE RELATIVE TO BEGINNING OF FILE
C
C     1.0 READ THE SCAN LINES FROM IMAGE FILE, CONVERT FROM BYTE TO INTEGER
C         AND STORE IN IARR()
C
      JL=0                              ! ROW INDEX IN STORAGE ARRAY
      DO 200 L=JL1,JL2
        JL=JL+1
        JRELL=JRELL+1
        CALL ZAWRLI(LUNID,JBH,JBL,JNP,JRELL,JRELP,JWD,QQQ,
     +    IERR,IOFS,BUF)
        IF(IERR.NE.1)THEN
          WRITE(5,20)IERR,L
20        FORMAT(1X//' ERROR NO.',I4,' REPORTED BY ZAWRLI',/,
     +      ' WHEN READING LINE',I5,'    DATA EXTRACTION BY RWIND',
     +      ' ABORTED.',//)
          GOTO 500
        END IF
C
        JP=0                            ! COLUMN INDEX IN STORAGE FILE
        JDUM=0
        DO 100 P=IOFS,IOFS+JWD-1        ! EXTRACT PIXEL COUNTS
```

```
            JP=JP+1
            BDUM(1)=BUF(P)                    ! CONVERT TO INTEGER VIA EQUIVALENCE
            IARR(JL,JP)=JDUM
100       CONTINUE
200     CONTINUE
C
C       2.0 RETURN
C
500     RETURN
        END
```

```
C
C       SUBROUTINE RTWIN.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       RTWIN IS CALLED BY MT2COM TO ROTATE THE TEMPLATE WINDOW DATA PRIOR
C       TO COMPUTATION OF THE MAXIMUM CORRELATION FUNCTION.
C       THE PROCESS CONSISTS OF :
C        (A) CALCULATION OF 4 COORDINATE PAIRS : LINE/PIXEL FOR ROTATED WINDOW
C            AND CORRESPONDING LINE/PIXEL BEFORE ROTATION.
C        (B) USING ARIES II MODULE ZSPOLY TO COMPUTE THE COEFFICIENTS FOR A
C            PAIR OF FIRST ORDER POLYNOMIAL TRANSFORM EQUATIONS.
C        (C) USING THE TRANSFORM EQUATIONS TO DETERMINE THE COORDINATES IN THE
C            NONROTATED IMAGE AND OBTAIN A NEW PIXEL VALUE USING THE WEIGHTED
C            MEAN OF THE NEIGHBOURS (BILINEAR INTERPOLATION REALY).
C
C       PROVISION IS MADE FOR A MAXIMUM WINDOW SIZE OF 17X17 AND ROTATION ANGLE
C       90 TO -90 DEGREES. ROTATION IS AROUND THE WINDOW CENTRE.
C
C       RTWIN(JCL,JCP,JBH,JBL,JNP,JFL,JFP,JWD,LUNID,SROT,QQQ,IERR,IARR)
C       JCL,JCP     INT*2  WINDOW CENTRE LINE AND PIXEL
C       JBH,JBL       ,,   HIGH AND LOW ORDER BYTES OF THE BASE ADDRES FOR
C                          THE TEMPLATE IMAGE DISK FILE.
C       JNP           ,,   NUMBER OF PIXELS PER LINE IN THE TEMPLATE FILE.
C       JFL,JFP       ,,   FIRST LINE AND PIXEL IN TEMPLATE IMAGE FILE.
C       JWD           ,,   WINDOW DIMENSION
C       LUNID         ,,   LOGICAL UNIT NUMBER FOR THE IMAGE DISK
C       SROT        REAL*4 WINDOW ROTATION ANGLE (DEGREES) .
C       IERR        INT*2  ERROR CODE. 1=NO ERROR
C       IARR(17,17)   ,,   ARRAY CONTAINING THE ROTATED WINDOW DATA.
C
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE RTWIN(JCL,JCP,JBH,JBL,JNP,JFL,JFP,JWD,LUNID,SROT,
     + QQQ,IERR,IARR)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION IARR(17,17),SX1(4),SY1(4),SX2(4),SY2(4),SXCOEF(10),
     + SYCOEF(10),BVAL(4),BUF(1024),BDUM(2)
      EQUIVALENCE (BDUM(1),JDUM)
      DATA BVAL/'T','T','T','T'/
C
      IERR=1
      JH=JWD/2                    ! HALF WINDOW SIZE
      VH=FLOAT(JH)
      VCL=FLOAT(JCL)
      VCP=FLOAT(JCP)
      VL1=VCL-VH                  ! FIRST LINE IN WINDOW
      VP1=VCP-VH                  ! FIRST PIXEL IN WINDOW
      VL2=VCL+VH                  ! LAST LINE IN WINDOW
      VP2=VCP+VH                  ! LAST PIXEL IN WINDOW
      VR=SQRT(2.0*VH*VH)          ! DISTANCE FROM CENTRE TO CORNER
      RPI=4.0*ATAN(1.0)           ! PI
      RDR=RPI/180.0               ! DEGREES TO RADIANS CONVERSION
      VRAD=RDR*SROT               ! ROTATION ANGLE IN RADIANS
      SWF=SQRT(2.0)
```

```
C
C       1.0 COMPUTE THE COORDINATE PAIRS. THE RESAMPLING IS CARRIED OUT IN THE
C           REVERSE DIRECTION IE. IF L2,P2 ARE THE LINE/PIXEL COORDINATES IN TH
C           WINDOW AFTER ROTATION THEN WE WISH TO COMPUTE L1,P1 FOR THE POINT
C           WHICH HAD BEEN ROTATED TO POSITION L2,P2. CONSEQUENTLY A CLOCKWISE
C           ROTATION IS TREATED AS AN ANTI-CLOCKWISE ROTATION AND VV.
C
        SX1(1)=VP1
        SX1(2)=VP2
        SX1(3)=VP2
        SX1(4)=VP1
        SY1(1)=VL1
        SY1(2)=VL1
        SY1(3)=VL2
        SY1(4)=VL2
C
        DO 100 J=1,4
          VX2=SX1(J)-VCP
          VY2=VCL-SY1(J)
          VTH=ATAN2(VX2,VY2)
          IF(VTH.LT.0.0)VTH=VTH+2.0*RPI
          VTH=VTH-VRAD
          VX1=VR*SIN(VTH)
          VY1=VR*COS(VTH)
          SX2(J)=VX1+VCP
          SY2(J)=VCL-VY1
100     CONTINUE
C       WRITE(6,120)((SY1(J),SX1(J),SY2(J),SX2(J)),J=1,4)
C120    FORMAT(1X/,4(1X,F8.3,2X,F8.3,5X,F9.4,2X,F9.4,/))
C
C       2.0 ZERO THE SXCOEF() AND SYCOEF() ARRAYS AND CALL ARIES II MODULE
C           ZSPOLY TO COMPUTE THE NEW COEFFICIENTS.
C
        DO 200 J=1,10
          SXCOEF(J)=0.0
          SYCOEF(J)=0.0
200     CONTINUE
C
        CALL ZSPOLY(1,4,BVAL,SX1,SY1,SX2,SY2,QQQ,IERR,SXCOEF,SYCOEF)
C       WRITE(6,210)(SYCOEF(J),J=1,5),(SXCOEF(J),J=1,5)
C210    FORMAT(1X/' SYCOEF=',5F12.6,/,' SXCOEF=',5F12.6)
        IF(IERR.NE.1)THEN
         WRITE(5,220)IERR
220      FORMAT(1X//' ERROR NO.',I5,' REPORTED BY ZSPOLY',/,
     +   ' ROTATION OPERATION BY RTWIN ABORTED',//)
         GOTO 500
        END IF
C
C       3.0 USE THE COEFFICIENTS PRODUCED BY ZSPOLY TO COMPUTE NEW PIXEL
C           COUNTS. FOR EACH PIXEL IN THE WINDOW 4 PIXELS ARE READ FROM THE
C           TEMPLATE IMAGE FILE AND THE WEIGHTED MEAN COMPUTED AT L1,P1.
C
        JP=0
        DO 400 VP=VP1,VP2
          JP=JP+1                              ! COLUMN INDEX FOR STORAGE ARRAY
          JL=0
          DO 380 VL=VL1,VL2
            JL=JL+1                            ! ROW INDEX FOR STORAGE ARRAY.
C
C           3.1 DETERMINE COORDINATES OF THE 4 PIXELS FROM WHICH THE MEAN IS
```

```
C                TO BE CALCULATED
C
            VX1=SXCOEF(1)+SXCOEF(2)*VP+SXCOEF(3)*VL
            VY1=SYCOEF(1)+SYCOEF(2)*VP+SYCOEF(3)*VL
            L1=INT(VY1)
            L2=L1+1
            P1=INT(VX1)
            P2=P1+1
            SY2(1)=FLOAT(L1)
            SY2(2)=SY2(1)
            SY2(3)=FLOAT(L2)
            SY2(4)=SY2(3)
            SX2(1)=FLOAT(P1)
            SX2(2)=FLOAT(P2)
            SX2(3)=SX2(1)
            SX2(4)=SX2(2)
C
C           3.2 READ THE 4 VALUES FROM THE DISK FILE USING ARIES II MODULE
C               ZAWRLI.
C
            JRELP=P1-JFP+1                  ! PIXEL RELATIVE TO BEGINNING OF LINE
            JRELL=L1-JFL+1                  ! LINE      ,,     ,,      ,,     ,, FILE
            J1=0
            DO 280 L=JRELL,JRELL+1
              CALL ZAWRLI(LUNID,JBH,JBL,JNP,L,JRELP,2,QQQ,IERR,IOFS,BUF)
              IF(IERR.NE.1)THEN
                WRITE(5,240)IERR,L
240             FORMAT(1X//' ERROR NO.',I5,' REPORTED BY ZAWRLI',/,
     +            ' READING RELATIVE LINE NO.',I5,/,
     +            ' ROTATION OPERATION BY RTWIN ABORTED.',//)
                GOTO 500
              END IF
C
              DO 260 P=IOFS,IOFS+1          ! EXTRACT THE 2 PIXEL COUNTS
                J1=J1+1
                JDUM=0
                BDUM(1)=BUF(P)
                SX1(J1)=FLOAT(JDUM)         ! STORE THE PIXEL COUNT.
260           CONTINUE
280         CONTINUE
C           IF(JL.LE.5)WRITE(6,290)JL,JP,VL,VP,VY1,VX1,SY2,SX2,SX1
C290        FORMAT(1X/' JL,JP,VL,VP,VY1,VX1=',2I5,2F9.3,2F11.4,/,
C    +      ' SY2=',4F7.2,'   SX2=',4F7.2,/,' SX1=',4F7.1)
C
C           3.3 CALCULATE THE NEW VALUE FROM THE 4 NEIGHBOURS.
C
            RXF=VX1-SX2(1)
            RYF=VY1-SY2(1)
            IARR(JL,JP)=INT(RXF*RYF*(SX1(1)-SX1(2)-SX1(3)+SX1(4))+
     +      RXF*(SX1(2)-SX1(1))+RYF*(SX1(3)-SX1(1))+SX1(1)+0.5)
380       CONTINUE
400     CONTINUE
C
C       4.0 RETURN
C
500     RETURN
        END
```

```
>
C
C       SUBROUTINE CRCOEF.FTN
C       A ROUTINE CREATED FOR USE WITH PROGRAM ADVECT - AUTOMATIC FEATURE
C       TRACKING FOR SEA SURFACE ADVECTION VECTORS.
C      MODIFIED SLIGHTLY FOR USE WITH PROGRAM MTRACK - MANUAL FEATURE TRACKING
C
C       CRCOEF DO TEMPLATE MATCHING BETWEEN TWO INTEGER ARRAYS, THE TEMPLATE
C       ITARR(17,17) AND THE ARRAY TO BE SEARCHED, ISARR(27,27).
C       THE BEST POSITION OF THE TEMPLATE WITHIN THE SEARCH ARRAY IS DETERMINE
C       BY THE HIGHEST CORRELATION COEFFICIENT RMAXCC.
C       THE COR. COEFFICIENT IS CALCULATED ACCORDING TO EQUATION NO.1 IN
C       TABLE 1 ,PAGE 143 OF SVEDLOV ET.AL. 1978.
C       THE ABSULUTE VALUE OF THE CORR. COEFF. IS <=1.0 . THIS SUBROUTINE
C       RETURNS THE LARGEST CORR. COEFF. ALONG WITH ITS ROW AND COLUMN
C       NUMBERS. NOTE !! : IF MORE THAN ONE POSITION YIELD THE SAME - LARGEST
C       COEFF. THEN THE FIRST POSITION ENCOUNTERED WILL BE RETURNED.
C
C       !!! NB !!!  DIMENSIONS OF BOTH THE TEMPLATE AND THE SEARCH ARRAY
C       SHOULD BE ODD.
C
C
C       J.J.AGENBAG  S.F.R.I.  JUNE 1990.  FORTRAN 77
C      MODIFIED APRIL 1991 FOR USE WITH MTRACK
C
C
       SUBROUTINE CRCOEF(N,M,ITARR,ISARR,XXX,RMAXCC,IROW,ICOL)
       IMPLICIT COMPLEX(A),BYTE(B),CHARACTER*1(C),INTEGER*4(E)
       DIMENSION RCSTOR(17,2),RCWORK(17,2),RTSUM(3),RTWORK(2),RXY(2),
     + ITARR(17,17),ISARR(27,27)
C
C
       RN2=FLOAT(N*N)              ! N=DIMENSI@N OF TEMPLATE ARRAY.
       IS=1+N/2            ! LEFT-MOST AND TOP-MOST POSITION OF THE CENTRE OF
                           ! THE TEMPLATE IN THE SEARCH ARRAY
       IE=M-N/2            ! M=DIMENSION OF SEARCH ARRAY. RIGHT-MOST AND BOTTOM-
                           ! MOST POSITION OF THE CENTRE OF THE TEMPLATE.
       IP=N/2
C
C      1.0 DO SUMMATIONS OF TEMPLATE ARRAY. THESE SUMS WILL BE CONSTANT
C          FOR THIS TEMPLATE.
C
       SUMX=0.0            ! SUM OF ALL TEMPLATE ELEMENTS.
       SUMX2=0.0           ! SUM OF SQUARES OF TEMPLATE ELEMENTS.
       DO 100 I=1,N        ! THE ROWS
         DO 80 J=1,N       ! THE COLUMNS
           SUMX=SUMX+FLOAT(ITARR(J,I))
           SUMX2=SUMX2+(FLOAT(ITARR(J,I)))**2.0
80         CONTINUE
100      CONTINUE
       RC=RN2*SUMX2-(SUMX)**2.0          ! A CONSTANT IN THE COR. COEFF.
C
C
C      CALCULATION OF THE CORRELATION COEFFICIENTS FOR ALL POSITIONS OF
C      THE TEMPLATE WITHIN THE SEARCH ARRAY :
C      POSITION THE TEMPLATE IN THE TOP LEFT CORNER OF THE SEARCH ARRAY.
C      THEN CALCULATE THE FOLLOWING SUMS FOR THE NXN ARRAY UNDER THE TEMPLATE.
C        1. SUMY= SUM OF SEARCH ARRAY ELEMENTS. (X=TEMPLATE, Y=SEARCH ARRAY)
C        2. SUMY2=SUM OF Y**2
```

```
C          3. SUMXY=SUM OF X*Y
C        THE SUMS ARE CALCULATED FOR EACH COLUMN AND STORED IN RCSTOR(N,3)
C        AS WELL AS FOR THE TOTAL ARRAY RTSUM(3). RCSTOR IS COPIED TO
C        RCWORK(N,3) AND RTSUM() TO RTWORK().
C        THE COR. COEFF. IS CALCULATED FOR THIS POSITION. THE TEMPLATE IS THEI
C        SHIFTED TOWARDS THE RIGHT - ONE COLUMN AT A TIME. FOR EACH POSITION
C        THE NEW SUMS ARE OBTAINED BY DISCARDING THE SUMS FOR THE LEFT-MOST
C        COLUMN ( SUBTRACT RCWORK(1,) FROM RTWORK() ) AND BY ADDING A NEW
C        SET OF SUMS FOR THE NEW RIGHT-MOST COLUMN.  ! NOTE. THIS CAN NOT BE
C        DONE TO GET SUMXY. FOR SUMXY THE INTIRE WINDOW MUST BE SUMMED.
C        WHEN REACHING THE END OF THE ROW (THE TEMPLATE CENTRE AT IE) THEN
C        THE TEMPLATE IS SHIFTED IN ROW DOWN AND BACK TO THE LEFT (TEMPLATE
C        CENTRE AT IS). THE COLUMN SUMS IN RCSTOR() AND THE TOTAL SUMS,
C        RTSUM(), ARE UPDATED BY SUBTRACTION OF THE VALUES IN THE DISCARDED
C        TOP-MOST ROW AND BY ADDITION OF THE VALUES IN THE NEW BOTTOM ROW.
C        ! ONCE AGAIN  RTSUM(3) (=SUMXY) MUST BE CALCULATED FROM A FULL
C        SUMMATION OVER THE NEW WINDOW.
C
C
C        2.0 INITIALISE FOR TOP LEFT CORNER.
C
         DO 140 I=1,2              ! 1=SUMY,2=SUMY2 AND 3=SUMXY
           DO 120 J=1,N            ! THE COLUMNS
             RCSTOR(J,I)=0.0       ! ZERO THE COLUMN SUMS.
120        CONTINUE
           RTSUM(I)=0.0            ! ZERO THE OVERALL SUMS
140      CONTINUE
         RTSUM(3)=0.0
C
         RMAXCC=0.0       ! INITIALISE THE MAX. CORR. COEFF.
C
C
C        2.1  FOR THE TOP LEFT POSITION OBTAIN THE COLUMN SUMS. DO THE OVERALI
C             SUMS AT THE SAME TIME
C
         DO 200 J=1,N              ! THE ROWS
           DO 160 I=1,N            ! THE COLUMNS
             RXY(1)=FLOAT(ISARR(I,J))              ! =Y
             RXY(2)=RXY(1)**2.0                    ! =Y**2
             DO 156 K=1,2          ! 1=SUMY,2=SUMY2
               RCSTOR(I,K)=RCSTOR(I,K)+RXY(K)
               RTSUM(K)=RTSUM(K)+RXY(K)
156          CONTINUE
             RTSUM(3)=RTSUM(3)+RXY(1)*FLOAT(ITARR(I,J))
160        CONTINUE
200      CONTINUE
C
C        3.0 FOR THE LEFT-MOST POSITION IN THE ROW CALCULATE THE COR. COEFF.
C
         IR1=1    ! TOP ROW IN SEARCH ARRAY (FOR THIS WINDOW POSITION)
         IR2=N    ! BOTTOM ROW IN SEARCH ARRAY
         IR=IS    ! ROW POSITION OF CENTRE OF TEMPLATE ( IN ISARR() AND RCORC(
210      IC=IS    ! COLUMN POSITION OF CENTRE OF TEMPLATE.
         JC=N+1
         RSQTF=RC*(RN2*RTSUM(2)-RTSUM(1)**2.0)
         IF(RSQTF.LE.0.0)GOTO 215
         RCORC=(RN2*RTSUM(3)-SUMX*RTSUM(1))/SQRT(RSQTF)
         IF(ABS(RCORC).GT.ABS(RMAXCC))THEN
           RMAXCC=RCORC
           IROW=IR
```

```
          ICOL=IC
        END IF
C
C       4.0 TRANSFER THE CONTENTS OF RCSTOR() TO RCWORK() AND RTSUM() TO
C           TO RTWORK(). SUBTRACT COLUMN 1 SUMS FROM RTWORK()
C
215     DO 240 J=1,2
          RTWORK(J)=RTSUM(J)
          RTWORK(J)=RTWORK(J)-RCSTOR(1,J)
          DO 220 I=2,N
            RCWORK(I-1,J)=RCSTOR(I,J)
220       CONTINUE
240     CONTINUE
C
C       5.0 SHIFT THE WINDOW TOWARDS THE RIGHT - ONE COLUMN AT A TIME.
C
250     DO 360 IC=IS+1,IE
C         5.1 DO A NEW N'TH COLUMN SUMMATION
          RCWORK(N,1)=0.0
          RCWORK(N,2)=0.0
          K=0
          RTSUM(3)=0.0
          DO 280 I=IR1,IR2                ! THE ROWS IN THE COLUMN
            K=K+1
            RXY(1)=FLOAT(ISARR(JC,I))     ! Y
            RXY(2)=RXY(1)**2.0            ! Y**2
            DO 260 J=1,2
              RCWORK(N,J)=RCWORK(N,J)+RXY(J)
              RTWORK(J)=RTWORK(J)+RXY(J)
260         CONTINUE
C
C           5.1.1 SUM OVER THE ENTIRE WINDOW (ROWS IR1-IR2,COLUMS IC-IP TO
C                 IC+IP) FOR RTSUM(3) - IE SUMXY
C
            L=0
            DO 270 J=IC-IP,IC+IP
              L=L+1
              RTSUM(3)=RTSUM(3)+FLOAT(ISARR(J,I))*FLOAT(ITARR(L,K))
270         CONTINUE
280       CONTINUE
C
C         5.2 CALCULATE THE CORR. COEFF.
C
          RSQTF=RC*(RN2*RTWORK(2)-RTWORK(1)**2.0)
          IF(RSQTF.LE.0.0)GOTO 300
          RCORC=(RN2*RTSUM(3)-SUMX*RTWORK(1))/SQRT(RSQTF)
          IF(ABS(RCORC).GT.ABS(RMAXCC))THEN
            RMAXCC=RCORC
            IROW=IR
            ICOL=IC
          END IF
C
C         5.3 IF NOT AT THE END OF THE ROW THEN SUBTRACT COLUMN 1 SUMS FROM
C             RTWORK() AND SHIFT THE COLUMNS ONE OVER.
C
300       IF(IC.EQ.IE)GOTO 360            ! END OF THE ROW.
          DO 340 J=1,2
            RTWORK(J)=RTWORK(J)-RCWORK(1,J)
            DO 320 I=2,N
              RCWORK(I-1,J)=RCWORK(I,J)
```

```
320           CONTINUE
340         CONTINUE
            JC=JC+1
360       CONTINUE
C
C
C         6.0 AT THE END OF A ROW. GO DOWN ONE ROW AND BACK TO THE START POINT.
C             SUBTRACT ROW IR1, Y AND Y**2 VALUES FROM THE RCSTOR() AND
C             RTSUM(). ADD THOSE FOR THE NEW LAST ROW -IR2+1
C
          IR=IR+1
          IF(IR.GT.IE)GOTO 500              ! NO MORE ROWS. DONE.
          IR2=IR2+1         ! INCREMENT WINDOW BOTTOM ROW INDEX.
          DO 400 I=1,N
            RXY(1)=FLOAT(ISARR(I,IR2))-FLOAT(ISARR(I,IR1))
            RXY(2)=(FLOAT(ISARR(I,IR2)))**2.0-(FLOAT(ISARR(I,IR1)))**2.0
            DO 380 K=1,2
              RCSTOR(I,K)=RCSTOR(I,K)+RXY(K)
              RTSUM(K)=RTSUM(K)+RXY(K)
380         CONTINUE
400       CONTINUE
          IR1=IR1+1         ! INCREMENT WINDOW TOP ROW INDEX.
C
C         6.1 DO A NEW SUMMATION FOR SUMXY OVER ROWS IR1-IR2 AND COLUMNS 1-N
C
          RTSUM(3)=0.0
          L=0
          DO 440 I=IR1,IR2
            L=L+1
            DO 420 J=1,N
              RTSUM(3)=RTSUM(3)+FLOAT(ISARR(J,I))*FLOAT(ITARR(J,L))
420         CONTINUE
440       CONTINUE
C
          GOTO 210
C
C
C
500       RETURN
          END
```

```
C
C       SUBROUTINE MTRAC3.FTN
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       THIS SUBROUTINE READS VECTOR DATA WRITTEN BY MTRAC2 TO A DIRECT ACCESS
C       STORAGE FILE (CREATED BY MTRAC1). A RECORD IN THIS FILE CONSIST OF:
C
C       N CL1 CP1 CL2 CP2 VL1 VP1 VL2 VP2          WHERE
C
C       N = A SEQUENTIAL VECTOR NUMBER ( =0 WHEN VECTOR DELETED )
C       (CL1,CP1) AND (CL2,CP2) = THE LINE/PIXEL COORDINATES OF THE VECTOR
C                                 START AND END POINTS AS INITIALLY DEFINED.
C       (VL1,VP1) AND (VL2,VP2) = COORDINATES FOR THE VECTOR AS DRAWN BY
C                                 MTRAC2 (DIFFER FROM CL1,CP1 ETC. IF A
C                                 SCALING FACTOR (#1) WAS APPLIED OR THE
C                                 VECTOR CENTRED OVER THE START POINT(SFAC,CPOS)
C
C       (CL1,CP1) AND (CL2,CP2) ARE PASSED TO SUBROUTINE LISTV WHICH COMPUTES
C       THE ACTUAL SPEED AND DIRECTION OF THE VECTOR FROM THE TRANSFORM DATA
C       IN RTRANS() AND LISTS THE RESULTS ON THE PRINTER.
C
C       ON COMPLETION THE VECTORS (IN THEME FIELD) ARE SAVED TO THE IMAGE FILE.
C
C       CALL MTRAC3(CIMF,IRNUM,IRSAV,LUNDF,RTRANS,IDISP,IDAREA,ITAREA)
C      CIMF           CHAR*1  MAIN MENU FUNCTION
C      IRNUM          INT*2   NUMBER OF DATA RECORDS IN THE STORAGE FILE.
C      IRSAV          INT*2   NUMBER OF RECORDS WHEN DATA LAST SAVED BY MTRAC3
C      LUNDF          INT*2   LOGICAL UNIT NUMBER ASSIGNED TO THE DATA FILE.
C      RTRANS(6,2)    REAL*4  IMAGE GEOMETRIC TRANSFORM DATA (,1)= TEMPLATE IMAGE,
C                             (,2)= SEARCH IMAGE DISPLAY
C                             (1,)&(2,)= A REFERENCE POINT LAT. AND LONG.
C                             (3,)&(4,)= LINE/PIXEL NUMBERS FOR THE REFERENCE PT.
C                             (5,)= SCALING FACTOR (NO. OF PIXELS=1 MIN OF LONG)
C                             (6,1)= TIME LAG BETWEEN IMAGES (HOURS)
C                             (6,2)= PRINTER INITIALISATION FLAG (0= NOT DONE)
C      IDISP(2,5)     INT*2   (1,1) & (2,1) = TEMPLATE IMAGE AND FEATURE NO.
C                             (1,2) & (2,2) = SEARCH   IMAGE AND FEATURE NO.
C                             (1,3) & (2,3) = VECTOR IMAGE AND THEME NUMBER
C                             (1,4) & (2,4) = TEMPLATE DISPLAY START LINE AND PIXEL
C                             (1,5) & (2,5) = SEARCH IMAGE DISPLAY START LINE/PIXEL
C      IDAREA(2,2)    INT*2   ENTIRE IMAGE AREA. (1,)&(2,)=LINE AND PIXEL NUMBERS
C                             (,1)=TOP LEFT CORNER; (,2)=BOTTOM RIGHT CORNER.
C      ITAREA(2,2)    INT*2   PART OF IMAGE TO SAVE. (1,) & (2,)= LINE AND PIXEL
C                             (,1)=TOP LEFT CORNER; (,2)=BOTTOM RIGHT CORNER.
C
C       J.J.AGENBAG  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE MTRAC3(CIMF,IRNUM,IRSAV,LUNDF,RTRANS,IDISP,IDAREA,
     + ITAREA)
C
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W)
      DIMENSION RTRANS(6,2),JVC(2,2),KVC(2,2),IDISP(2,5),IDAREA(2,2),
     + ITAREA(2,2)
      PARAMETER (PR=6)
      IERR=1
C
```

```
C
40      WRITE(5,50)
50      FORMAT(1X//'  LIST ONLY      (L)',/,
     + '  SAVE ONLY      (S)',/,
     + '  LIST AND SAVE (B)',//,
     + '  SELECT (L,S,B,CTRL-Z)(B):',$)
        CALL KEYBD(CVAL,COUT,JVAL)          ! READ KEYBOARD INPUT.
        IF(CVAL.EQ.'Z')GOTO 220             ! CTRL-Z. RETURN TO MAIN PROGRAM.
        IF(CVAL.EQ.'E')COUT='B'             ! ACCEPT DEFAULT.
        IF(COUT.NE.'L'.AND.COUT.NE.'S'.AND.COUT.NE.'B')GOTO 40    ! TRY AGAIN.
        CF1=COUT
C
55      WRITE(5,60)
60      FORMAT(1X/' LIST/SAVE ALL VECTORS (A) OR ONLY THOSE DEFINED',/,
     + ' SINCE THE LAST LISTING/SAVING (L) ?',//,
     + ' SELECT (A,L,CTRL-Z)(A) :',$)
        CALL KEYBD(CVAL,COUT,JVAL)                  ! READ KEYBOARD INPUT
        IF(CVAL.EQ.'Z')GOTO 40                      ! CTRL-Z. STEP BACK.
        IF(CVAL.EQ.'E')COUT='A'                     ! ACCEPT DEFAULT.
        IF(COUT.NE.'A'.AND.COUT.NE.'L')GOTO 55      ! TRY AGAIN
        CF2=COUT
C
        IF(CF1.EQ.'S')GOTO 110                      ! SAVE VECTORS. NO LISTING.
C
        RTRANS(6,2)=0                       ! SET FLAG TO PRINT HEADER
        N=0
        J1=1
        IF(CF2.EQ.'L')J1=IRSAV+1
        DO 100 J=J1,IRNUM                   ! READ RECORDS
          READ(UNIT=LUNDF,REC=J,FMT=80)M,JVC,KVC
80        FORMAT(1X,9I6)
          IF(M.EQ.0)GOTO 100                ! VECTOR BEEN ERASED
          DO 90 J1=1,2
           DO 84 J2=1,2
            IF(JVC(J2,J1).EQ.0)GOTO 100
            IF(KVC(J2,J1).EQ.0)GOTO 100
84         CONTINUE
90        CONTINUE
          CALL LISTV(JVC,RTRANS,N)
100     CONTINUE
        IF(COUT.EQ.'L')GOTO 200             ! DO NOT SAVE VECTORS.
C
110     WRITE(5,120)
120     FORMAT(1X//' SAVING VECTORS ON DISK. PLEASE WAIT.')
        IF(CF2.EQ.'A')THEN
          CALL ZIUNTH(IDISP(1,3),IDAREA(1,1),IDAREA(2,1),IDAREA(1,2),
     +    IDAREA(2,2),QQQ,IERR)
         ELSE
          CALL ZIUNTH(IDISP(1,3),ITAREA(1,1),ITAREA(2,1),ITAREA(1,2),
     +    ITAREA(2,2),QQQ,IERR)
        END IF
        IF(IERR.NE.1)THEN
          WRITE(5,140)IERR
140       FORMAT(1X//' ERROR NO.',I4,' REPORTED BY ZIUNTH IN SUBROUTINE ',
     +    'MTRAC3.',/)
          GOTO 220
        END IF
        IRSAV=IRNUM
C
200     CIMF='X'
```

```
220     RETURN
        END
```

```
C
C       SUBROUTINE LISTV.FTN
C
C       PART OF PROGRAM MTRACK - MANUAL FEATURE TRACKING.
C
C       THE SUBROUTINE USES INFORMATION ABOUT THE TRANSFORMATION TO MERCATOR
C       PROJECTION TO COMPUTE THE GEOGRAPHICAL COORDINATES (LATS/LONGS)
C       CORRESPONDING WITH THE IMAGE COORDINATES (LINE/PIXEL) OF THE VECTOR
C       AND FROM THESE COMPUTES THE SPEED AND DIRECTION OF THE VECTOR IN
C       ABSOLUTE UNITS. RESULTS ARE LISTED ON THE PRINTER.
C
C     CALL LISTV(IVC,RTRANS,IVNUM)
C     IVC(2,2)       INT*2  CURSOR COORDINATES : (1,)=LINE   (2,)=PIXEL
C                           (,1)=TEMPLATE (,2)=SEARCH IMAGE DISPLAY
C     RTRANS(6,2)    REAL*4 IMAGE GEOMETRIC TRANSFORM DATA (,1)= TEMPLATE IMAG
C                           (,2)= SEARCH IMAGE DISPLAY
C                           (1,)&(2,)= A REFERENCE POINT LAT. AND LONG.
C                           (3,)&(4,)= LINE/PIXEL NUMBERS FOR THE REFERENCE PT
C                           (5,)= SCALING FACTOR (NO. OF PIXELS=1 MIN OF LONG)
C                           (6,1)= TIME LAG BETWEEN IMAGES (HOURS)
C                           (6,2)= PRINTER INITIALISATION FLAG (0= NOT DONE)
C     IVNUM          INT*2  VECTOR SEQUENTIAL NUMBER
C
C
C       J.J.AGENEAU  S.F.R.I.  APRIL 1991. IN FORTRAN 77 AND USING ARIES II
C       IMAGE PROCESSING SOFTWARE MODULES. FOR A DEC LSI 11/23 BASED SYSTEM
C
C
      SUBROUTINE LISTV(IVC,RTRANS,IVNUM)
      IMPLICIT BYTE(B),CHARACTER*1(C),INTEGER*4(E),INTEGER*2(I-P),
     + REAL*4(Q-W),REAL*8(X-Z)
      DIMENSION IVC(2,2),RTRANS(6,2),SLL(2,2)
      PARAMETER (PR=6)
C
      XPI=4.0D0*DATAN(1.0D0)                ! PI
      RPI=XPI
      XDR=XPI/180.0D0                       ! DEGREES TO RADIAN CONVERSION.
      RDR=XDR
      SEQR=3963.3                          ! EARTH EQUATORIAL RADIUS (NAUTICAL MI
      SPOR=3949.8                          ! EARTH POLAR RADIUS            ,,
      SECC=SQRT((SEQR-SPOR)*(SEQR+SPOR))/SEQR  ! ECCENTRICITY OF EARTH.
      SCALE=(2.0*RPI*SEQR)/(360.0*60.0)
C
C     1.0 IF REQUIRED, PRINT HEADER
C
      IF(RTRANS(6,2).EQ.0.0)THEN
       WRITE(PR,20)
20     FORMAT(1X/1X,'NO.  LIN1 PIX1  LIN2 PIX2    LAT1     LON1',
     +  '      LAT2     LON2     CM/S KNOT  DIRECT',/)
       RTRANS(6,2)=1.0
      END IF
C
C     2.0 COMPUTE LATS AND LONGS. STORE IN SLL().
C
      DO 200 J=1,2                  ! 1=START POINT, 2=END POINT
        SLL(2,J)=(RTRANS(4,J)-FLOAT(IVC(2,J)))/(60.0*RTRANS(5,J))+
     +   RTRANS(2,J)                ! LONGITUDE
        ASSIGN 50 TO JCO
        SLAT=RTRANS(1,J)
```

```
        GOTO 500                ! GOTO COMMON ROUTINE.
50      SD1=SDIST               ! DISTANCE TO EQUATOR FROM REF. LAT
C
        ASSIGN 60 TO JGO
        SLAT=RTRANS(1,J)+1.0            ! LAT. 1 DEG NORTH OF REFERENCE
        GOTO 500                        ! GOTO COMMON ROUTINE
60      SD2=SDIST
        SF1=1.0/(RTRANS(5,J)*(SD2-SD1))  ! APPROXIMATE NUMEER OF DEGREES OF
                                         ! LAT. EQUAL TO 1 LINE.
C
C       2.1 ITERATE TO FIND THE LATITUDE FOR LINE IVC(1,J)
C
        JIC=0                   ! ITERATION STEP COUNTER
        SLIN=FLOAT(IVC(1,J))
        SL=RTRANS(1,J)-(SLIN-RTRANS(3,J))*SF1    ! APPROXIMATE LATITUDE.
        SLN=SL+1.0              ! NORTHERN ITERATION LIMIT
        SLS=SL-1.0              ! SOUTHERN      ,,      ,,
        ASSIGN 100 TO JGO
80      SLAT=(SLN+SLS)/2.0      ! TRY THE MID POINT
        GOTO 500                ! GO TO COMMON ROUTINE
C
100     JIC=JIC+1
        SLINE=RTRANS(3,J)+RTRANS(5,J)*(SD1-SDIST) ! LINE NO. FOR LAT SLAT
        IF(ABS(SLINE-SLIN).GT.0.01)THEN           ! MATCH NOT GOOD ENOUGH
          IF(SLINE.LT.SLIN)SLN=SLAT         ! ADJUST LIMITS
          IF(SLINE.GT.SLIN)SLS=SLAT         !   ,,     ,,
          IF(JIC.GE.20)GOTO 600             ! GIVE UP.
          GOTO 80                           ! NEXT ITERATION
        END IF
        SLL(1,J)=SLAT                       ! THE REQUIRED LATITUDE
200   CONTINUE
C
C     3.0 CALCULATE THE DIRECTION AND DISTANCE (KM) BETWEEN THE TWO POINTS
      XD1=(SLL(1,2)-SLL(1,1))*XDR           ! LATITUDINAL DISTANCE. RADIANS
      XD2=(SLL(2,1)-SLL(2,2))*XDR           ! LONGITUDINAL     ,,     ,,
      XAD2=DABS(XD2)
      IF(XD1.EQ.0.0D0.AND.XD2.EQ.0.0D0)THEN
        SPEED1=0.0
        SPEED2=0.0
        XDIR=0.0
        XDIST=0.0
        GOTO 300
      END IF
C
      XL=(SLL(1,1)+SLL(1,2))*XDR/2.0            ! MEAN LATITUDE. RADIANS
      XR=6378.137D0*(1.0D0-0.00335D0*(DSIN(XL))**2.0) ! EARTH RADIUS (KM)
C
      IF(XD1.EQ.0.0D0)XD1=0.000000001D0
      XDIR=(DATAN2(XD2,XD1))/XDR                ! DIRECTION IN DEGREES
      IF(XDIR.LT.0.0D0)XDIR=XDIR+360.0D0
C
      XLN=(90.0-SLL(1,1))*XDR
      XLS=(90.0-SLL(1,2))*XDR
      XDIST=DACOS(DCOS(XLN)*DCOS(XLS)+DSIN(XLN)*DSIN(XLS)*DCOS(XAD2))
      XDIST=XDIST*XR                            ! DISTANCE IN KM.
      SPEED1=(XDIST*1000.0)/(RTRANS(6,1)*36.0)  ! SPEED IN CM/S
      SPEED2=SPEED1/51.4785                     ! SPEED IN KNOTS
C
C     4.0 WRITE THE RESULTS.
C
```

```
300     IVNUM=IVNUM+1
        WRITE(PR,320)IVNUM,IVC,SLL,SPEED1,SPEED2,XDIR
320     FORMAT(1X,I3,I6,I5,I6,I5,F9.2,F8.2,F8.2,F8.2,F8.1,F5.2,F8.2)
        GOTO 600
C
C ------------------------------------------------------------------------
C
C       5.0 COMMON ROUTINE TO COMPUTE FOR A MERCATOR PROJECTION THE DISTANCE
C           (IN MINUTES OF LONGITUDE) FROM THE EQUATOR TO A GIVEN LATITUDE.
C
500     SRLAT=SLAT*RDR                                  ! LATITUDE IN RADIANS
        STAN=TAN((RPI/4.0)+(SRLAT/2.0))
        SSIN=SIN(SRLAT)
        SFAC=((1.0-SECC*SSIN)/(1.0+SECC*SSIN))**(SECC/2.0)
        SDIST=SEQR*LOG(STAN*SFAC)/SCALE
        GOTO JGO
C
C ------------------------------------------------------------------------
C
C
600     RETURN
        END
```

```
PIP>T; I:=MTRAC.CMD      CK.CMD
;
;       TASK BUILD INDIRECT COMMAND FILE FOR MTRACK - DERIVE ADVECTION VECTORS
;       THROUGH MANUAL FEATURE TRACKING.
;
 MTRACK.TSK/FP/CP=MTRACK.ODL/MP
;
 COMMON=DPICOM:RW:5
 UNITS=6
 ASG=DD1:1,IM:2,SY:3,SY:4,TI:5,TT1:6
;
;       GET REMAINING INORMATION FROM THE OVERLAY DESCRIPTOR FILE MTRACK.ODL
;       END
;


PIP>TI:=MTRACK.ODL
;
;      OVERLAY DESCIPTOR FILE FOR MTRACK - DERIVE ADVECTION VECTORS THROUGH
;      MANUAL FEATURE TRACKING.
;
        .ROOT MTRACK-KEYBD-*(A,B,C,D)
;
 A:     .FCTR MTRAC0
;
;
 B:     .FCTR MTRAC1-LB1-LB2-(B1,B2,B3)
 B1:    .FCTR READPT-LB2
 B2:    .FCTR SELFEA-LB3
 B3:    .FCTR SELVD
;
;
 C:     .FCTR MTRAC2-DANDD-LB2-(C1,C2,C3,C4,C5)
;
 C1:    .FCTR MT2MEN-LB2
;
 C2:    .FCTR MT2DMS-(C21,C22)
 C21:   .FCTR MT2VC1-CWIND1-LB2
 C22:   .FCTR MT2AR1-CARR01
;
 C3:    .FCTR MT2ROT-CWIND4
;
 C4:    .FCTR MT2COM-(C41,C42,C43,C44)
 C41:   .FCTR MT2VC2-CWIND2-LB2
 C42:   .FCTR MT2AR2-CARR02
 C43:   .FCTR RTWIN-RWIND-LB3
 C44:   .FCTR CRCOEF
;
 C5:    .FCTR MT2AEX-(C51,C52)
 C51:   .FCTR MT2VC3-CWIND3-LB2
 C52:   .FCTR MT2AR3-CARR03
;
;
 D:     .FCTR MTRAC3-LISTV-LB2
;
 LB1:   .FCTR DR:[100,300]NONRLS/LB
 LB2:   .FCTR DR:[100,300]VMALIB/LB
 LB3:   .FCTR DR:[100,300]COMMON/LB
;
  .END
```